MVR Series
Development Kit

Programmer's Manual

Version 3.50J

■ご購入製品を使用される際の注意事項

ここでは、プログラムを行うハードウェア（MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000）をご使用になられるときにご注意いただきたい事項について説明しています。ハードウェアのご使用方法や、本注意事項の内容についてご不明な点、疑問点などございましたら、カノープス株式会社テクニカルサポートまでお問い合わせください。
MVR Series Development Kit のサポートは、カノープスシステム開発サポート（p.19 参照）で承っております。

カノープス株式会社
〒651-2241
神戸市西区室谷 1-2-2（神戸ハイテクパーク内）
テクニカルサポート TEL.078-992-6830
（祝祭日および当社指定休日を除く月～金 10:00～12:00　13:00～17:00）

●ご利用目的に関する注意事項

医療機器や人命に関るシステムでは、絶対にご利用にならないでください。製品の性質上、これらのシステムへの導入は適しません。

●製品の取り付けおよび取り外しに関する注意事項

製品の取り付けや取り外しを行う場合、必ずパソコン本体および周辺機器の電源を切り、さらに電源ケーブルをコンセントから抜いた状態で行ってください。
パソコン本体および周辺機器の電源を入れたまま製品を取り付けたり取り外したりした場合、製品やパソコン本体、周辺機器および周辺機器に接続されている機器の一部が破壊される恐れがあります。 また、パソコン本体および周辺機器の電源ケーブルをコンセントから抜かずにパソコン本体や周辺機器の筐体（電源ユニットなど）、機器の金属部分を触った場合には感電する恐れがあります。

●静電気に関する注意事項

製品に静電気が流れると製品上の部品が破壊される恐れがあります。各コネクタや部品面（MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000）には直接手を触れないでください。
静電気は衣服や人体からも発生します。製品に触る前に、一旦接地された金属製のものに触れてください（体内の静電気を放電することになります）。

●消費電流に関する注意事項

複数の拡張ボードをパソコンに取り付けるときは、ご購入製品を含めた全ての製品の消費電流の合計がパソコンの最大供給電流を越えていないことを必ず確認してください。全ボードの消費電流の合計がパソコンの最大供給電流を越えたりするなどの動作条件を満たさない環境で使用し続けると、システムが正常に動作しない場合やシステムに負荷がかかり、パソコンが故障する原因となる恐れがあります。

消費電流のわからない製品については、その製品の取扱説明書をご覧いただくか、メーカーに直接お問い合わせいただいてお確かめください。

●他社製品との併用に関する注意事項

他社製品と併用されるとご購入製品は正常に動作しないことがあり、そのためにシステムが本来の目的を達成することができないこともあります。あらかじめ、製品単体の環境でご購入製品が正常に動作することをご確認ください。また、他社製品との併用でご購入製品が正常に動作しないのであれば、その他社製品とご購入製品との併用はお止めください。

●その他の注意事項

製品は指定された位置に指示通り取り付けてください。指示通りに取り付けられてない場合、製品（MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000）の金属部とパソコンの金属部が接触してショートするなどの要因で、製品やパソコン本体・周辺機器が破壊される恐れがあります。

製品を取り扱うときは手など皮膚を傷つけないよう十分にご注意ください。ハードウェアの仕様上、製品のパネル、コネクタ、エッジ、裏面は金属のピンが突出していることがあります。製品を取り付けたり取り外したりするときは、製品全体を軽く包み込むようにお持ちください。

動作中の製品は熱により非常に熱くなります。長時間使用した製品に手を触れる際には、十分ご注意ください。

ご注意

(1)本製品の一部または全部を無断で複製することを禁止します。
(2)本製品の内容や仕様は将来予告無しに変更することがあります。
(3)本製品は内容について万全を期して作成いたしましたが、万一ご不審な点や誤り、記載漏れなどお気付きの事がございましたら、当社までご連絡ください。
(4)運用した結果については、(3)項にかかわらず責任を負いかねますのでご了承ください。
(5)本製品付属のソフトウェア、マニュアル、その他添付物を含めたすべての関連品に関して、解析、リバースエンジニアリング、デコンパイル、ディスアッセンブリを禁じます。
(6)カノープス、Canopus およびそのロゴはカノープス株式会社の登録商標です。MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000 および MVR Series Development Kit はカノープス株式会社の商標です。
(7)MS、Windows は米国マイクロソフト・コーポレーションの登録商標です。またその他の商品名やそれに類するものは各社の商標または登録商標です。

表記について

■本書では Microsoft® Windows® operating system を単に『Windows』、Microsoft® WindowsNT® operating system を単に『WindowsNT』と表記します。
■本書は Windows 上でプログラムを作成することができる方を対象に書かれています。
■本書での説明と実際の運用方法とで相違点がある場合には、実際の運用方法を優先するものとします。プログラム上の基本的な事項、プログラムの方法などについては記載していません。
■本書に記載されていない情報が記載される場合がありますので、ディスクに添付のテキストファイルも必ずお読みください。

目次

CHAPTER 0　Preliminaries

1. 関数の仕様変更

Version1.10 から Version2.00 への変更点

- **Version1.10** から **Version2.00** へのアップデートにより、次の関数について引数の変更や構造体のメンバー追加など仕様が変更になりました。詳細については、**CHAPTER3** および **CHAPTER4** の各関数についての説明をご覧ください。

関数	変更内容
ENC_Set_Callback	引数の **ENC_CB_TMAP** が **ENC_VOBU_ENT** に変わりました。また、新たに引数は2つ追加されました。
ENC_Initialize	引数の **ENC_MEDIA** が削除されました。**ENC_MEDIA** の取得／設定については、**ENC_Get_Media** 関数／**ENC_Set_Media** 関数で行ないます。
ENC_Can_Overlay_Window	引数が1つ追加されました。
ENC_Create_Overlay_Window	引数が2つ追加されました。
ENC_Get_Video_Encoder_Parameter_Ex	**ENC_VIDEO_ENCODE_PARAMETER_EX** 構造体の **reserved** メンバーが削除され、新たに3つのメンバーが追加されました。
ENC_Set_Video_Encoder_Parameter_Ex	同上
ENC_Init_Movie	引数の **ENC_MODE** が削除されました。
DEC_Can_Overlay_Window	引数が1つ追加されました。
DEC_Create_Overlay_Window	引数が2つ追加されました。
DEC_Get_Decode_Parameter	**DEC_DECODE_PARAMETER** 構造体にメンバーが2つ追加されました。
DEC_Set_Decode_Parameter	同上

※お試し版で追加された関数で、仕様が変更になったものについては記述しておりません。

Version2.00 から Version2.10 への変更点

Version2.00 から **Version2.10** へのアップデートにより、仕様が変更になりました。変更内容は以下のとおりです。

- 次の関数について引数の変更や構造体のメンバー追加など仕様が変更になりました。詳細については、**CHAPTER3** および **CHAPTER4** の各関数についての説明をご覧ください。

関数	変更内容
DEC_Get_Decode_Parameter	**DEC_DECODE_PARAMETER** 構造体にメンバーが1つ追加されました。
DEC_Set_Decode_Parameter	同上

- この開発キットは、マイクロソフト株式会社の **Visual C++ 4.2** 用から **Visual C++ 6.0** 用に変更になりました。

Version2.10 から Version3.00 への変更点

- **Version2.10** から **Version3.00** へのアップデートにより、次の関数について引数の変更や構造体のメンバー追加など仕様が変更になりました。詳細については、**CHAPTER3** および **CHAPTER4** の各関数についての説明をご覧ください。

関数	変更内容
ENC_Get_Video_Encoder_Parameter_Ex	**ENC_VIDEO_ENCODE_PARAMETER_EX** 構造体にメンバーが1つ追加されました。
ENC_Set_Video_Encoder_Parameter_Ex	同上

※新規追加された API については、ここでは記述しておりません。

2. 開発キットの内容

MVR Series Development Kit(以下、開発キットと記します)は、ビデオ映像を MPEG2/MPEG1 フォーマットでキャプチャしたり、その逆に MPEG2/MPEG1 フォーマットのファイルをビデオ信号として出力する **MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000** の機能をアプリケーションに組み込むためのコンポーネントです。

この開発キットには以下に示すように、インクルードファイルおよびサンプルプログラム(ソースコードと実行ファイル)で構成されています。

C/C++用インクルードファイル

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Include¥Mvrapidef.h

C/C++用ライブラリファイル

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Lib¥Mvrapi.lib

C/C++用サンプルプログラム(ソースコード)

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Samples¥Encode …エンコード

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Samples¥Encode Memory …エンコード(メモリ渡し)

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Samples¥Com¥Encode communication Ex …エンコード(通信)

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Samples¥Decode …デコード

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Samples¥Decode Memory …デコード(メモリ渡し)

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Samples¥Com¥Decode Communication Ex …デコード(通信)

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Samples¥Multi …複数枚制御

Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Samples¥MvrCtrl …ActiveX コントロール

C/C++用サンプルプログラム（実行ファイル）

Program Files¥Canopus¥MVR-D2000¥Encode.exe	…エンコード
Program Files¥Canopus¥MVR-D2000¥Encode_m.exe	…エンコード（メモリ渡し）
Program Files¥Canopus¥MVR-D2000¥Encode_cEx.exe	…エンコード（通信）
Program Files¥Canopus¥MVR-D2000¥Decode.exe	…デコード
Program Files¥Canopus¥MVR-D2000¥Decode_m.exe	…デコード（メモリ渡し）
Program Files¥Canopus¥MVR-D2000¥Decode_cEx.exe	…デコード（通信）
Program Files¥Canopus¥MVR-D2000¥Multi.exe	…複数枚制御
Program Files¥Canopus¥MVR-D2000¥MvrCtrl.ocx	…ActiveX コントロール

3. 開発キットのインストール

本開発キットのインストールは、MVR-D2200V ユーザーズマニュアル『2-5.ソフトウェアのインストール』を参考に、「コンポーネントの選択」ダイアログボックスから「Development Kit」を指定し、インストールを行ってください。

4. Visual C++でのアプリケーション開発

●開発環境

　プロジェクトで参照されるインクルードファイルおよびライブラリファイルのフォルダに以下のパスを追加してください。

ポイント：本書では開発キットが組み込まれているフォルダを **C：¥Program Files¥Canopus¥MVR-D2000 Development Kit¥**であると想定しています。お使いの環境に合わせて読み替えてください。

インクルードファイル

C：¥ Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Include

ライブラリ

C：¥ Program Files¥Canopus¥MVR-D2000 Development Kit¥VC¥Lib

●コンパイル

次のファイルをインクルードしてください。

Mvrapidef.h

●リンク

次のライブラリをリンクしてください。

Mvrapi.lib

5. 他の開発ツールでのアプリケーション開発

この開発キットはマイクロソフト株式会社の *Visual C++* 6.0 用です。

他の開発ツール(言語処理系)によるアプリケーションのコンパイル、リンクおよび実行結果に関するお問い合わせには十分にお応えできない場合があります。あらかじめご了承ください。

お使いの開発ツール(言語処理系)の設定や付属ユーティリティの使用により、*Visual C++* のインクルードファイルやライブラリを利用する方法については、開発ツールのメーカーにお問い合わせください。

6. サンプルプログラム

開発キットにはこのドキュメントに記載されているファンクションの具体的な使用例として *Visual C++* に対応したサンプルが含まれています。作成しようとしているアプリケーションに組み込む機能に応じて各サンプルを参照してください。

サンプルプログラムの内容は **C : ¥Program Files¥Canopus¥MVR-D2000 Development Kit¥SAMPLES.TXT**を参照してください。

ポイント：本書では開発キットが組み込まれているフォルダを C:¥Program Files¥Canopus¥MVR-D2000 Development Kit¥であると想定しています。お使いの環境に合わせて読み替えてください。

VC¥Samples¥Encode	エンコードプログラム
VC¥Samples¥Encode Memory	エンコード（メモリ渡し）プログラム
VC¥Samples¥Com¥Encode Communication Ex	エンコード（通信）プログラム
VC¥Samples¥Decode	デコードプログラム
VC¥Samples¥Decode Memory	デコード（メモリ渡し）プログラム
VC¥Samples¥Com¥Decode Communication Ex	デコード（通信）プログラム
VC¥Samples¥Multi	複数枚制御
VC¥Samples¥MvrCtrl	ActiveX コントロール

Visual C++ 用のサンプルは MFC を使用して記述しています。

7. 開発キットに含まれるソースコードの取り扱いについて

本開発キットに含まれるサンプルプログラムはドキュメントを補うための資料となっています。サンプルプログラムのソースコード（以下、「ソースコード」といいます）自体を改変したり、その一部をお客様のアプリケーションに組み込んで利用してください。ただし、お客様のアプリケーションとそこに組み込まれたソースコードの一部との適合性に関してサポートの範囲を限定させていただくこともございますのでご了承ください。

カノープス株式会社はソースコードの使用と変更に関して完全に自由な権利をお客様に許諾いたします。ただし、これらのコードをソースコードもしくはこれを変更したものの形態でお客様の営利を目的としたソフトウェア製品に含めることはご遠慮願います。

また、最終的に作成されたアプリケーションの運用結果および目的への適合性につき、カノープス株式会社では一切の責任を負いかねますので予めご了承ください。

8. プログラム作成時の注意事項

本開発キットに関するご質問は、インターネット E-mail または FAX でのみ承っております。お電話や NIFTY-Serve でのお問い合わせについては受付できませんのであらかじめご了承ください。

インターネット E-mail　mvrsdk@canopus.co.jp
カノープスシステム開発サポート　FAX：078-992-4203

お問い合わせの際には発生現象と共に次の内容を必ず記載してください。

1. 使用しているモジュールのバージョン情報
   ☆エクスプローラからファイルを右クリックすることで確認できます。
2. 使用している開発環境
   ・ご使用の Windows または Windows NT 環境（Service Pack のバージョン）
   ☆Windows NT 環境でお使いの場合は Service Pack 4 以降がインストールされている環境が必要です。
   ☆対応 OS 環境は、**MVR-D2200V**/MVR-D2200/MVR-D2000/MPL-D2000 本体の動作環境に準拠します。
   ☆Windows 3.1、Windows NT 3.5x ではお使いいただけません。（2003.6 現在）
   ・コンパイラのメーカー、バージョン、対応言語
   ・使用されているその他の開発キット
3. 使用している実行環境
   ・PC 本体メーカーおよび機種名
   ・CPU の種類
   ・搭載メモリ容量
   ・ご使用の Windows または Windows NT 環境（Service Pack のバージョン）
   ☆Windows NT 環境でお使いの場合は Service Pack 4 以降がインストールされている環境が必要です。
   ☆対応 OS 環境は、**MVR-D2200V**/MVR-D2200/MVR-D2000/MPL-D2000 本体の動作環境に準拠します。
   ☆Windows 3.1、Windows NT 3.5x ではお使いいただけません。（1999.9 現在）
   ・周辺機器メーカーおよび型番

CHAPTER 1　　Tutorial

1. エンコードアプリケーションの作成

エンコードアプリケーションでは、

カードの選択
エンコーダの使用開始
オーバーレイウィンドウの生成
ウィンドウの移動およびサイズ変更
エンコードの開始
エンコードの停止
オーバーレイウィンドウの破棄
エンコーダの使用終了

などの機能が必要となります。

この項では、ビデオ映像をオーバーレイウィンドウに表示してからエンコードを行うアプリケーションの作成について説明します。本文中ではエラーチェックは省いてあります。

※エンコードに関する機能は **MPL-D2000** では使用できません。

カードの選択

MVR-D2200V/MVR-D2200/MVR-D2000 を複数枚装着している場合、どのカードが使用可能かを調べ、使用可能なカードの中から使用するカードを指定します。
MVR-D2200V/MVR-D2200/MVR-D2000 を 1 枚だけ装着している場合や使用するカードを選ばない場合は、カードを指定する必要はありません。

```
UINT            enc_id;                 // エンコーダの識別子(グローバル変数)
        ENC_CONFIG      config;         // カード情報を取得するための構造体
        int             i;

        // カード枚数を取得
        memset(&config, 0, sizeof(ENC_CONFIG));
        ENC_Get_Codec_Config(enc_id, &config);

        // 使用できるカードを探す
        for (i = 1; i <= config.NumberCodecs; i++)
        {
                config.CodecNumber = i;
                ENC_Get_Codec_Config(enc_id, &config);
                if (config.CurrentUtilization == FALSE)
                {
                        if (config.CodecCaps & ENCCAP_ENCODE)
                        {
                                // 使用するカードを指定する
                                ENC_Set_Codec_Number(enc_id, i);
                                break;
                        }
                }
        }
```

エンコーダの使用開始

MVR-D2200V/MVR-D2200/MVR-D2000 を使用するアプリケーションは、その使用開始をドライバに許可してもらう必要があります。アプリケーションの初期化時に **ENC_Initialize** または **ENC_Initialize_Ex** ファンクションを呼び出して **MVR-D2200V/MVR-D2200/MVR-D2000** の使用を開始します。

ファイルへのエンコード時の例

```
UINT            enc_id;         // エンコーダの識別子(グローバル変数)
ENC_MEDIA       media;          // エンコードの種類(グローバル変数)
        ENC_RETURN      enc_return;

        enc_return = ENC_Initialize(enc_id);
        if (enc_return != ENC_SUCCESS)
        {
                // MVR-D2000 が使用できない

                エラー処理
        }

        media = ENC_MEDIA_FILE;
        enc_return = ENC_Set_Media(enc_id, media);
        if (enc_return != ENC_SUCCESS)
        {
                // エンコードの種類を設定できない

                エラー処理
        }
```

メモリ転送時の例

```
UINT                enc_id;             // エンコーダの識別子(グローバル変数)
ENC_MEDIA           media;              // エンコードの種類(グローバル変数)
        ENC_RETURN              enc_return;
        ENC_BSS_PARAMETER       bss_param;

        bss_param.fType         = ENC_BSS_TYPE_PS;
        bss_param.pfnCallback   = WriteProc;            // コールバック関数
        bss_param.cbBuff        = 0x8000;               // バッファサイズ
        bss_param.cBuff         = 16;                   // バッファ数
        bss_param.dwData        = (DWORD)INVALID_HANDLE_VALUE;

        media = ENC_MEDIA_MEMORY;
        enc_return = ENC_Initialize_Ex(enc_id, media, &bss_param);
        if (enc_return != ENC_SUCCESS)
        {
                // MVR-D2000 が使用できない

                エラー処理
        }

// メモリ転送用コールバック関数
void CALLBACK WriteProc(LPBYTE pbBuff, DWORD cbBuff, DWORD dwData)
{
        DWORD cbWritten;
        HANDLE hFile = (HANDLE)dwData;

        if (hFile != INVALID_HANDLE_VALUE)
        {
                WriteFile(hFile, pbBuff, cbBuff, &cbWritten, NULL);
        }
}
```

オーバーレイウィンドウの生成

アプリケーションのウィンドウ生成終了後、オーバーレイ表示が可能か調べてから、クライアント領域を **Overlay Window** として使用します。

```
UINT    enc_id;          // エンコーダの識別子(グローバル変数)
UINT    nWidth;          // オーバーレイ表示する幅(グローバル変数)
UINT    nHeight;         // オーバーレイ表示する高さ(グローバル変数)
HWND    hWndApp;         // アプリケーションのメインウィンドウハンドル(グローバル変数)
HWND    hWndOverlay;     // オーバーレイウィンドウのハンドル(グローバル変数)
        ENC_RETURN      enc_return;

        enc_return = ENC_Can_Overlay_Window(enc_id, NULL, nWidth, nHeight);
        if (enc_return != ENC_SUCCESS)
        {
                // オーバーレイ表示ができない

                エラー処理
                return;
        }

        enc_return = ENC_Create_Overlay_Window(enc_id, hWndApp, &hWndOverlay, 0,
0, nWidth, nHeight) ;
        if (enc_return != ENC_SUCCESS)
        {
                // オーバーレイウィンドウの生成に失敗した

                エラー処理
                return;
        }
```

ウィンドウの移動およびサイズ変更

ユーザーの操作により、アプリケーションのウィンドウを移動させた時や、ウィンドウのサイズを変更した場合には、オーバレイウィンドウの新しい位置への移動やサイズ変更を行ないます。

```
UINT    enc_id;         // エンコーダの識別子（グローバル変数）

case WM_MOVE:
        {
        INT     xPos, yPos;

        xPos = (INT)(short) LOWORD(lParam);
        yPos = (INT)(short) HIWORD(lParam);
        ENC_Move_Overlay_Window(enc_id, xPos, yPos) ;
        }
        return 0;

case WM_SIZE:
        {
        UINT    nWidth, nHeight;

        nWidth = (UINT)LOWORD(lParam);
        nHeight = (UINT)HIWORD(lParam);
        ENC_Resize_Overlay_Window(enc_id, nWidth, nHeight);
        }
        return 0;
```

エンコードの開始

エンコードを開始する。
エンコードを開始する前に必ず **ENC_Init_Movie** ファンクションを呼び出してエンコード開始待ち状態にする。その後で **ENC_Record_Movie** ファンクションを呼び出してエンコードを開始します。

```
UINT            enc_id;                 // エンコーダの識別子(グローバル変数)
ENC_MEDIA       media;                  // エンコードの種類(グローバル変数)
HANDLE          hFile;                  // ファイルハンドル
TCHAR           szFile[MAX_PATH];       // ファイル名
        ENC_BSS_PARAMETER   bss_param;

        if (media == ENC_MEDIA_MEMORY)
        {
                hFile = CreateFile(szFile, GENERIC_READ | GENERIC_WRITE,
                                FILE_SHARE_READ, NULL,
                                CREATE_ALWAYS,
                                FILE_ATTRIBUTE_NORMAL |
                                FILE_FLAG_SEQUENTIAL_SCAN,
                                NULL);

                ENC_Get_BSS_Parameter(enc_id, &bss_param);
                bss_param.dwData = (DWORD)hFile;
                ENC_Set_BSS_Parameter(enc_id, & bss_param);
        }

        ENC_Init_Movie(enc_id);
        ENC_Record_Movie(enc_id, ENC_RECORD_PS);
```

エンコードの停止

エンコード中に停止する場合は、**ENC_Stop** ファンクションを呼び出します。

```
UINT    enc_id;          // エンコーダの識別子（グローバル変数）

        ENC_Stop(enc_id);
```

オーバーレイウィンドウの破棄

オーバーレイウィンドウを破棄する。**WM_DESTROY** などで実行します。

```
UINT    enc_id;          // エンコーダの識別子（グローバル変数）

        ENC_Destroy_Overlay_Window(enc_id );
```

エンコーダの使用終了

MVR-D2200V/MVR-D2200/MVR-D2000 を使用するアプリケーションは、その使用終了をドライバに通知する必要があります。アプリケーションの終了時などに **ENC_Terminate** ファンクションを呼び出して **MVR-D2200V/MVR-D2200/MVR-D2000** を終了します。

```
UINT    enc_id;          // エンコーダの識別子（グローバル変数）

        ENC_Terminate(enc_id );
```

2. デコードアプリケーションの作成

デコードアプリケーションでは、

カードの選択
デコーダの使用開始
オーバーレイウィンドウの生成
ウィンドウの移動およびサイズ変更
デコードの開始
デコードの停止
オーバーレイウィンドウの破棄
デコーダの使用終了

などの機能が必要となります。

この項では、画像データをオーバーレイウィンドウに表示しながらデコードを行うアプリケーションの作成について説明します。本文中ではエラーチェックは省いてあります。

カードの選択

MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000 を複数枚装着している場合、どのカードが使用可能かを調べ、使用可能なカードの中から使用するカードを指定します。
MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000 を 1 枚だけ装着している場合や使用するカードを選ばない場合は、カードを指定する必要はありません。

```
UINT            dec_id;                 // デコーダの識別子(グローバル変数)
        DEC_CONFIG      config;         // カード情報を取得するための構造体
        int             i;

        // カード枚数を取得
        memset(&config, 0, sizeof(DEC_CONFIG));
        DEC_Get_Codec_Config(dec_id, &config);

        // 使用できるカードを探す
        for (i = 1; i <= config.NumberCodecs; i++)
        {
                config.CodecNumber = i;
                DEC_Get_Codec_Config(dec_id, &config);
                if (config.CurrentUtilization == FALSE)
                {
                        if (config.CodecCaps & DECCAP_DECODE)
                        {
                                // 使用するカードを指定する
                                DEC_Set_Codec_Number(dec_id, i);
                                break;
                        }
                }
        }
```

デコーダの使用開始

MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000 を使用するアプリケーションは、その使用開始をドライバに許可してもらう必要があります。アプリケーションの初期化時に **DEC_Initialize** または **DEC_Initialize_Ex** ファンクションを呼び出して **MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000** の使用を開始します。

ファイルからのデコード時の例

```
UINT            dec_id;         // デコーダの識別子(グローバル変数)
DEC_MEDIA       media;          // デコードの種類(グローバル変数)

        DEC_RETURN    dec_return;

        dec_return = DEC_Initialize(dec_id);
        if (dec_return != DEC_SUCCESS)
        {
                // MVR-D2000 が使用できない

                エラー処理
        }

        media = DEC_MEDIA_FILE;
        dec_return = DEC_Set_Media(dec_id, media);
        if (dec_return != DEC_SUCCESS)
        {
                // デコードの種類を設定できない

                エラー処理
        }
```

メモリ転送時の例

```
UINT    dec_id;            // デコーダの識別子(グローバル変数)
        DEC_RETURN          dec_return;
        DEC_BSR_PARAMETER   bsr_param;

        bsr_param.fType         = DEC_BSR_TYPE_PS;
        bsr_param.pfnCallback   = ReadProc;             // コールバック関数
        bsr_param.cbBuff        = 0x8000;               // バッファサイズ
        bsr_param.cBuff         = 32;                   // バッファ数
        bsr_param.dwData        = (DWORD)INVALID_HANDLE_VALUE;

        media = DEC_MEDIA_MEMORY;
        dec_return = DEC_Initialize_Ex(dec_id, media, &bsr_param);
        if (dec_return != DEC_SUCCESS)
        {
                // MVR-D2000 が使用できない

                エラー処理
        }

// メモリ転送用コールバック関数
void CALLBACK ReadProc(LPBYTE pBuff, DWORD cbWrite,
                                        LPDWORD pcbWritten, DWORD dwData)
{
        HANDLE hFile = (HANDLE)dwData;
        if (hFile != INVALID_HANDLE_VALUE)
        {
                ReadFile(hFile, pBuff, cbWrite, pcbWritten, NULL);
        }
}
```

オーバーレイウィンドウの生成

アプリケーションのウィンドウ生成終了後、オーバーレイ表示が可能か調べてから、クライアント領域を **Overlay Window** として使用します。

```
UINT    dec_id;          // デコーダの識別子(グローバル変数)
UINT    nWidth;          // オーバーレイ表示する幅(グローバル変数)
UINT    nHeight;         // オーバーレイ表示する高さ(グローバル変数)
HWND    hWndApp;         // アプリケーションのメインウィンドウハンドル(グローバル変数)
HWND    hWndOverlay;     // オーバーレイウィンドウのハンドル(グローバル変数)
        DEC_RETURN       dec_return;

        dec_return = DEC_Can_Overlay_Window(dec_id, NULL, nWidth, nHeight);
        if (dec_return != DEC_SUCCESS)
        {
                // オーバーレイ表示ができない

                エラー処理
                return;
        }

        dec_return = DEC_Create_Overlay_Window(dec_id, hWndApp, &hWndOverlay, 0,
0, nWidth, nHeight) ;
        if (dec_return != DEC_SUCCESS)
        {
                // オーバーレイウィンドウの生成に失敗した

                エラー処理
                return;
        }
```

ウィンドウの移動およびサイズ変更

ユーザーの操作により、アプリケーションのウィンドウを移動させた時や、ウィンドウのサイズを変更した場合には、オーバレイウィンドウの新しい位置への移動やサイズ変更を行ないます。

```
UINT    dec_id;           // デコーダの識別子(グローバル変数)

case WM_MOVE:
        {
        INT     xPos, yPos;

        xPos = (INT)(short) LOWORD(lParam);
        yPos = (INT)(short) HIWORD(lParam);
        DEC_Move_Overlay_Window(dec_id, xPos, yPos) ;
        }
        return 0;

case WM_SIZE:
        {
        UINT    nWidth, nHeight;

        nWidth = (UINT)LOWORD(lParam);
        nHeight = (UINT)HIWORD(lParam);
        DEC_Resize_Overlay_Window(dec_id, nWidth, nHeight);
        }
        return 0;
```

デコードの開始

デコードを開始するには、**DEC_Play** ファンクションを呼び出します。

```
UINT      dec_id;                          // デコーダの識別子(グローバル変数)
DEC_MEDIA       media;                     // デコードの種類(グローバル変数)
HANDLE          hFile;                     // ファイルハンドル
TCHAR           szFile[MAX_PATH];          // ファイル名
          DEC_BSR_PARAMETER   bsr_param;

          if (media == DEC_MEDIA_MEMORY)
          {
                    hFile = CreateFile(szFile, GENERIC_READ,
                                        FILE_SHARE_READ, NULL,
                                        OPEN_EXISTING,
                                        FILE_ATTRIBUTE_NORMAL |
                                        FILE_FLAG_NO_BUFFERING |
                                        FILE_FLAG_SEQUENTIAL_SCAN, NULL);

                    DEC_Get_BSR_Parameter(dec_id, &bsr_param);
                    bsr_param.dwData = (DWORD)hFile;
                    DEC_Set_BSR_Parameter(dec_id, &bsr_param);
          }

          DEC_Play(dec_id);
```

デコードの停止

デコード中に停止する場合は、**DEC_Stop** ファンクションを呼び出します。

```
UINT    dec_id;           // デコーダの識別子(グローバル変数)

        DEC_Stop(dec_id);
```

オーバーレイウィンドウの破棄

オーバーレイウィンドウを破棄します。**WM_DESTROY** などで実行します。

```
UINT    dec_id;           // デコーダの識別子(グローバル変数)

        DEC_Destroy_Overlay_Window(dec_id );
```

デコーダの使用終了

MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000 を使用するアプリケーションは、その使用終了をドライバに通知する必要があります。アプリケーションの終了時などに **DEC_Terminate** ファンクションを呼び出して **MVR-D2200V/MVR-D2200/MVR-D2000/MPL-D2000** を終了します。

```
UINT    dec_id;           // デコーダの識別子(グローバル変数)

        DEC_Terminate(dec_id );
```

CHAPTER 2 Quick Reference

§1. Video Encoder Functions

ENC_Set_Callback	コールバック関数の設定
ENC_Can_Initialize	エンコーダを初期化できるか調べる
ENC_Get_Codec_Config	カードに関する情報の取得
ENC_ Set_Codec_Number	使用するカード番号の設定
ENC_Initialize	エンコーダの初期化
ENC_Initialize_Ex	拡張エンコーダの初期化
ENC_Terminate	エンコーダの終了
ENC_Get_Media	エンコードの種類の取得
ENC_Set_Media	エンコードの種類の設定
ENC_Can_Record	録画／録音できるか調べる
ENC_Get_Status	ステータスの取得
ENC_Can_Overlay_Window	オーバーレイウィンドウの使用可能か調べる
ENC_Create_Overlay_Window	オーバーレイウィンドウの生成
ENC_Destroy_Overlay_Window	オーバーレイウィンドウの破棄
ENC_Move_Overlay_Window	オーバーレイウィンドウの移動
ENC_Resize_Overlay_Window	オーバーレイウィンドウのサイズ変更
ENC_Show_Overlay_Window	オーバーレイウィンドウの表示変更
ENC_Get_Overlay_Window	オーバーレイウィンドウのハンドル取得
ENC_Get_Overlay_Rect	オーバーレイウィンドウの表示領域の取得
ENC_Set_Overlay_Rect	オーバーレイウィンドウの表示領域の設定
ENC_Start_Monitor	モニタ開始
ENC_Stop_Monitor	モニタ停止
ENC_Get_Monitor_Status	モニタ状態の取得
ENC_Get_VideoCD_Mode	**Video CD** モードの取得
ENC_Set_VideoCD_Mode	**Video CD** モードの設定
ENC_Get_BSS_Parameter	メモリ転送用パラメータの取得
ENC_Set_BSS_Parameter	メモリ転送用パラメータの設定
ENC_Get_BSS_ParameterAV	エレメンタリストリームでのメモリ転送用パラメータを取得します。
ENC_Set_BSS_ParameterAV	エレメンタリストリームでのメモリ転送用パラメータを設定します。
ENC_Get_Overlay_Parameter	オーバーレイ表示パラメータの取得
ENC_Set_Overlay_Parameter	オーバーレイ表示パラメータの設定
ENC_Get_Video_Parameter	ビデオパラメータの取得
ENC_Set_Video_Parameter	ビデオパラメータの設定
ENC_Get_Video_Parameter2	ビデオパラメータ **2** の取得
ENC_Set_Video_Parameter2	ビデオパラメータ **2** の設定
ENC_Get_Video_Encode_Parameter	ビデオエンコードパラメータの取得
ENC_Set_Video_Encode_Parameter	ビデオエンコードパラメータの設定
ENC_Get_Video_Encode_Parameter_Ex	拡張ビデオエンコードパラメータの取得
ENC_Set_Video_Encode_Parameter_Ex	拡張ビデオエンコードパラメータの設定

ENC Get_Video_Encode_Parameter_Ex2	拡張ビデオエンコードパラメータ **2** の取得
ENC_Set_Video_Encode_Parameter_Ex2	拡張ビデオエンコードパラメータ **2** の設定
ENC_Get_Audio_Format	オーディオ形式の取得
ENC_Set_Audio_Format	オーディオ形式の設定
ENC_Get_Audio_Parameter	オーディオパラメータの取得
ENC_Set_Audio_Parameter	オーディオパラメータの設定
ENC_Get_Audio_Encode_Parameter	オーディオエンコードパラメータの取得
ENC_Set_Audio_Encode_Parameter	オーディオエンコードパラメータの設定
ENC_Init_Movie	エンコードの開始待ち状態にする
ENC_Record_Movie	エンコードの開始
ENC_Stop	エンコードの停止
ENC_Get_BitmapBits	静止画**(BMP)**をバッファで取得します。
ENC_Save_DIB	静止画**(BMP)**をファイルに保存します。
ENC_Save_JPEG	静止画**(JPEG)**をファイルに保存します。
ENC_Get_Record_Time	エンコード時間の取得
ENC_Set_Record_Time	エンコード時間の設定
ENC_Get_Movie_File	エンコードファイルの取得
ENC_Set_Movie_File	エンコードファイルの設定
ENC_Get_Frame_Count	エンコードを行なったフレーム数の取得
ENC_Get_Time	エンコードを行なった時間の取得
ENC_Detect_Video_Input_Source	入力ソースの自動検出
ENC_Set_Digital_Video_Inpug	アナログ・ビデオ入力とデジタル・ビデオ入力を切り換えます。
ENC_Get_Digital_Video_Inpug	現在のビデオ入力の状態を取得します。
ENC_Set_Video_Encode_File	詳細ビデオ・エンコード・パラメータ用ファイルの設定
ENC_Get_IRE_Setting	NTSC セットアップレベル（黒レベル）の設定を取得します。
ENC_Set_IRE_Setting	NTSC セットアップレベル（黒レベル）を設定します。
ENC_Get_Last_Error	エラーの取得

※エンコードに関する機能は **MPL-D2000** では使用できません。

§2. Video Decoder Functions

DEC_Get_Repeat	リピート状態の取得
DEC_Set_Repeat	リピート状態の設定
DEC_Get_Movie_File	デコードファイルの取得
DEC_Set_Movie_File	デコードファイルの設定
DEC_Get_Image_Size	画像サイズの取得
DEC_Get_Frame_Count	デコードを行なったフレーム数の取得
DEC_Get_Time	デコードを行なった時間取得
DEC_Get_Type	デコードタイプの取得
DEC_Get_File_Type	ファイルタイプの取得
DEC_Get_Playback_Time	再生時間の取得
DEC_Seek	再生開始位置の指定
DEC_Get_Sync_Stc_Value	**HD814210 の SYNC STC レジスタの値を取得**
DEC_Get_IRE_Setting	NTSC セットアップレベル（黒レベル）の設定を取得します。
DEC_Set_IRE_Setting	NTSC セットアップレベル（黒レベル）を設定します。
DEC_Get_Overlay_Parameter_Ex	オーバーレイの設定を取得します。
DEC_Set_Overlay_Parameter_Ex	オーバーレイの設定を行ないます。
DEC_Get_Overlay_Parameter_Range_Ex	オーバーレイ表示パラメータの範囲を取得します。
DEC_Get_Last_Error	エラーの取得

CHAPTER 3　Video Encoder Functions

ENC_Set_Callback

コールバック関数の設定を行ないます。

書式

ENC_RETURN ENC_Set_Callback(UINT enc_id, ENC_CB_STATUS pStatus,
ENC_CB_ERROR pError,
ENC_CB_VOBU_ENT pVobu_ent,
ENC_CB_S_PTM pSptm,
ENC_CB_SVOB_ENT pSvob_ent);

引数

enc_id	エンコーダの識別子
pStatus	内部状態通知用コールバック関数を設定する
pError	エラー通知用コールバック関数を設定する
pVobu_ent	**GOP** 完結モードでの情報取得用コールバック関数を設定する
pSptm	予約 **NULL** を渡す
pSvob_ent	予約 **NULL** を渡す

ENC_CB_STATUS コールバック関数の定義

エンコーダの内部状態を取得するためのコールバック関数

VOID CALLBACK StatusProc(UINT enc_id, ENC_STATUS_NOTIFY status);

enc_id	エンコーダの識別子
status	内部状態

値	説明
ENC_NOTIFY_INITIALIZE	エンコーダの初期化が行われた
ENC_NOTIFY_TERMINATE	エンコーダが終了した
ENC_NOTIFY_STOP	エンコードが停止した
ENC_NOTIFY_INIT_MOVIE	エンコード開始待ち状態になった
ENC_NOTIFY_MOVIE	エンコードを開始した

ENC_CB_ERROR コールバック関数の定義

エンコード中にエラーが発生した場合に、そのエラー情報を取得するためのコールバック関数

VOID CALLBACK ErrorProc(UINT enc_id, ULONG error);

enc_id	エンコーダの識別子
error	エラーコード（**APPENDIX** 1．エラーコード表参照）

ENC_CB_VOBU_ENT コールバック関数の定義

GOP 完結モードでの情報を取得するためのコールバック関数

```
VOID CALLBACK VobuProc(UINT enc_id, ENC_VOBU_ENT_INFO* pVobu_ent,
                                        UINT Boundary_Flag);
```

enc_id	エンコーダの識別子
pVobu_ent	**ENC_VOBU_ENT_INFO** 構造体へのポインタ
Boundary_Flag	常に **0** が返されます。

ENC_VOBU_ENT_INFO 構造体の定義

GOP 完結モードでの情報が格納される。

```
typedef struct {
        USHORT      Fstref_Sz;
        USHORT      Vobu_Pb_Tm;
        ULONG       Vobu_Sz;
} ENC_VOBU_ENT_INFO;
```

Fstref_Sz	完結単位中の最初の **I** ピクチャのセクタ**(PCK)**数。オーディオ・パック**(A_PCK)**を含む。
Vobu_Pb_Tm	完結単位の再生時間**(**単位**:**ビデオ・フィールド数**)**
Vobu_Sz	完結単位のセクタ**(PCK)**数

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

- **ENC_CB_ERROR** で定義したコールバック関数は、エンコード中にエラーが発生した場合に実行されます。関数を実行して **ENC_FAIL** が戻り値として返された場合は、**ENC_Get_Last_Error** によりエラー情報を取得してください。

参照

ENC_Get_Last_Error

ENC_Can_Initialize

エンコーダを初期化できるか調べます。

書式

ENC_RETURN ENC_Can_Initialize(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

エンコーダの初期化が可能な場合	**ENC_SUCCESS**
すでに使用しているかエンコーダを初期化できない場合	**ENC_FAIL**

ENC_Get_Codec_Config

カードに関する情報を取得します。

書式

ENC_RETURN ENC_Get_Codec_Config (UINT enc_id , ENC_CONFIG *pConfig);

引数

enc_id	エンコーダの識別子
pConfig	情報を取得する **ENC_CONFIG** 構造体へのポインタ

ENC_CONFIG 構造体の定義

カードに関する情報を取得する。

```
typedef struct {
    UINT    NumberCodecs;
    UINT    CodecNumber;
    UINT    CodecCaps;
    UINT    CurrentUtilization;
} ENC_CONFIG;
```

NumberCodecs 装着されているカード枚数が格納される。
CodecNumber **0** または情報を取得したいカード番号（**1**～**NumberCodecs**）を指定する。
CodecCaps **CodecNumber** で指定したカードに関する詳細情報フラグが格納される。

値	意味
ENCCAP_NONE	情報がないことを示すフラグ
ENCCAP_ENCODE	エンコードが可能であることを示すフラグ
ENCCAP_DECODE	デコードが可能であることを示すフラグ
ENCCAP_MONITOR	モニタが可能であることを示すフラグ
ENCCAP_AVI2MPEG	**AVI** ファイルを **MPEG** に変換可能であることを示すフラグ
ENCCAP_D2200	**MVR-D2200** であることを示すフラグ
ENCCAP_D2200V	**MVR-D2200** であることを示すフラグ

CurrentUtilization **CodecNumber** で指定したカードが使用状況が格納される。

値	意味
TRUE	使用している
FALSE	使用していない

戻り値

成功 **ENC_SUCCESS**

失敗　　　　　　　　**ENC_FAIL**

補足

ENC_CONFIG 構造体の **CodecNumber** に **0** を指定すると、装着されているカード枚数と現在使用しているカード番号を取得できます。カード枚数は **NumberCodecs** に格納されます。カード番号は **CodecNumber** に格納されます。カードが使用されていない場合は **CodecNumber** に **0** が格納されます。

ENC_Set_Codec_Number

使用するカード番号を設定します。

書式

ENC_RETURN ENC_Set_Codec_Number(UINT enc_id , UINT nNumber);

引数

enc_id	エンコーダの識別子
nNumber	使用するカード番号

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

- **ENC_Initialize** または **ENC_Initialize_Ex** による初期化を行う前に **ENC_Set_Codec_Number** を実行してください。初期化を行う前に **ENC_Set_Codec_Number** を実行しない場合は、自動的に使用可能なカードを探して初期化を行います。
- 指定できるカード番号は、**1** から **ENC_Get_Codec_Config** で取得した **ENC_CONFIG** 構造体の **NumberCodecs** までの値です。
- カード番号に **0** を指定した場合は、初期化時に自動的に使用可能なカードを探します。

参照

ENC_Get_Codec_Config, ENC_Initialize, ENC_Initialize_Ex

ENC_Initialize

エンコーダの初期化を行ないます。エンコードの種類は、ファイルへのエンコードになります。

書式

ENC_RETURN ENC_Initialize(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Initialize_Ex

エンコーダの初期化を行ないます。同時にメディアの設定とメモリ転送時の設定を行います。

書式

ENC_RETURN ENC_Initialize_Ex(UINT enc_id, ENC_MEDIA media,
ENC_BSS_PARAMETER* pParam);

引数

enc_id エンコーダの識別子
media エンコードの種類

値	意味
ENC_MEDIA_FILE	ファイルへのエンコード
ENC_MEDIA_MEMORY	メモリ転送

pParam メモリ転送用パラメータを設定する **ENC_BSS_PARAMETER** 構造体のポインタ
pParam は、**media** が **ENC_MEDIA_MEMORY** の場合に設定する。
それ以外の場合は、**NULL** を指定する。

ENC_BSS_PARAMETER 構造体の定義

メモリ転送用パラメータ情報を設定する。

```
typedef struct {
	ENC_BSS_TYPE		fType;
	ENC_CB_BSS		pfnCallback;
	DWORD			cbBuff;
	DWORD			cBuff;
	DWORD			dwData;
} ENC_BSS_PARAMETER;
```

fType 予約 **ENC_BSS_TYPE_PS** を渡す

ENC_CB_BSS コールバック関数の定義

エンコードデータを転送するためのコールバック関数

VOID CALLBACK BssProc(LPBYTE pbBuff, DWORD cbBuff, DWORD dwData);

pbBuff	エンコードデータが格納されているバッファのポインタ
cbBuff	エンコードデータのサイズ
dwData	**ENC_BSS_PARAMETER** 構造体で設定された **dwData** の値

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

- エレメンタリストリームでメモリ転送を行なう場合の設定は、**ENC_Set_BSS_Parameter_AV** を使用してください。
- **Video CD** モードでのメモリ転送は行なえません。

参照

ENC_Set_VideoCD_Mode, ENC_Set_BSS_Parameter_AV

ENC_Terminate

エンコーダを終了します。

書式

ENC_RETURN ENC_Terminate(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

●**ENC_Terminate** 実行後、**ENC_Set_Callback** によるコールバックの設定は全て解除されます。

参照

ENC_Set_Callback

ENC_Get_Media

エンコードの種類を取得します。

書式

ENC_RETURN ENC_Get_Media(UINT enc_id, ENC_MEDIA* media);

引数

enc_id	エンコーダの識別子
media	エンコードの種類を取得する **ENC_MEDIA** へのポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Initialize_Ex

ENC_Set_Media

エンコードの種類を設定します。

書式

ENC_RETURN ENC_Set_Media(UINT enc_id, ENC_MEDIA media);

引数

enc_id	エンコーダの識別子
media	エンコードの種類

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Initialize_Ex

ENC_Can_Record

エンコードが可能か判断します。

書式

ENC_RETURN ENC_Can_Record(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

エンコード可能	**ENC_SUCCESS**
エンコード不可能	**ENC_FAIL**

ENC_Get_Status

エンコーダの内部状態を取得します。

書式

ENC_RETURN ENC_Get_Status(UINT enc_id, ENC_STATUS* status);

引数

enc_id エンコーダの識別子
status エンコーダの内部状態を取得する **ENC_STATUS** へのポインタ

値	意味
ENC_STATUS_WAITINITIALIZE	初期化待ち状態
ENC_STATUS_WAITTERMINATE	終了待ち状態
ENC_STATUS_STOP	停止中
ENC_STATUS_MOVIE	エンコード中

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Can_Overlay_Window

オーバーレイウィンドウを使用できるかどうか判断します。

書式

ENC_RETURN ENC_Can_Overlay_Window(UINT enc_id, HANDLE hMonitor, UINT nWidth, UINT nHeight);

引数

enc_id	エンコーダの識別子
hMonitor	モニタハンドル
nWidth	オーバーレイウィンドウの幅
nHeight	オーバーレイウィンドウの高さ

戻り値

オーバーレイウィンドウが使用できる場合は、**ENC_SUCCESS**

オーバーレイウィンドウが使用できなければ、**ENC_FAIL**

補足

- **hMonitor** には、マルチモニタでのプライマリまたはセカンダリのモニタハンドルを設定します。マルチモニタでない場合は、**hMonitor** に **NULL** を設定します。

ENC_Create_Overlay_Window

オーバーレイウィンドウを生成します。

書式

ENC_RETURN ENC_Create_Overlay_Window(UINT enc_id, HWND hWndParent, HWND* hWndOverlay, int x, int y, UINT nWidth, UINT nHeight);

引数

enc_id	エンコーダの識別子
hWndParent	オーバーレイウィンドウを生成するウィンドウのハンドル
hWndOverlay	オーバーレイウィンドウのハンドルを取得するアドレス
x	オーバーレイウィンドウの位置（左）
y	オーバーレイウィンドウの位置（上）
nWidth	オーバーレイウィンドウの幅（最大 **720**）
nHeight	オーバーレイウィンドウの高さ（最大 **NTSC 480**、**PAL 576**）

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

- オーバーレイウィンドウ生成時は、サイズが制限されますが、生成後 **ENC_Resize_Overlay_Window** 行により制限以上のサイズを指定できます。
- エンコード中は生成できません。

参照

ENC_Resize_Overlay_Window

ENC_Destroy_Overlay_Window

オーバーレイウィンドウを破棄します。

書式

ENC_RETURN ENC_Destroy_Overlay_Window(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

●エンコード中は破棄できません。

ENC_Move_Overlay_Window

オーバーレイウィンドウを移動します。

書式

ENC_RETURN ENC_Move_Overlay_Window(UINT enc_id, INT x, INT y);

引数

enc_id	エンコーダの識別子
x	オーバーレイウィンドウの左上隅のスクリーン **x** 座標
y	オーバーレイウィンドウの左上隅のスクリーン **y** 座標

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Resize_Overlay_Window

オーバーレイウィンドウのサイズを変更します。

書式

ENC_RETURN ENC_Resize_Overlay_Window(UINT enc_id, UINT nWidth, UINT nHeight);

引数

enc_id	エンコーダの識別子
nWidth	オーバーレイウィンドウの幅
nHeight	オーバーレイウィンドウの高さ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Show_Overlay_Window

オーバーレイウィンドウの表示状態を切り換えます。

書式

ENC_RETURN ENC_Show_Overlay_Window(UINT enc_id, BOOL fShow);

引数

enc_id エンコーダの識別子
fShow オーバーレイウィンドウの表示／非表示

値	説明
TRUE	表示
FALSE	非表示

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Get_Overlay_Window

オーバーレイウィンドウのハンドルを取得します。

書式

ENC_RETURN ENC_Get_Overlay_Window(UINT enc_id, HWND* pWnd);

引数

enc_id	エンコーダの識別子
pWnd	オーバーレイウィンドウのハンドルを取得するアドレス

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Get_Overlay_Rect

オーバーレイウィンドウの表示領域を取得します。

書式

ENC_RETURN ENC_Get_Overlay_Rect(UINT enc_id, LPRECT pRect);

引数

enc_id	エンコーダの識別子
pRect	オーバーレイウィンドウの表示領域を取得する **RECT** 構造体へのポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Set_Overlay_Rect

オーバーレイウィンドウの表示領域を設定します。

書式

ENC_RETURN ENC_Set_Overlay_Rect(UINT enc_id, LPRECT pRect);

引数

enc_id	エンコーダの識別子
pRect	オーバーレイウィンドウの表示領域を設定した **RECT** 構造体へのポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

- **pRect** で設定可能な範囲は次のとおりです。

ビデオフォーマット	水平位置	垂直位置
NTSC	**0 ～ 720**	**0 ～ 480**
PAL	**0 ～ 720**	**0 ～ 576**

- エンコード中は設定できません。

ENC_Start_Monitor

モニタを開始します。

書式

ENC_RETURN ENC_Start_Monitor(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

●**ENC_Start_Monitor** を実行すると、オーバーレイウィンドウは生成時のサイズに戻ります。

●エンコード中は開始できません。

ENC_Stop_Monitor

モニタを停止します。

書式

ENC_RETURN ENC_Stop_Monitor(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

●エンコード中は停止できません。

ENC_Get_Monitor_Status

モニタの状態を取得します。

書式

ENC_RETURN ENC_Get_Monitor_Status(UINT enc_id, UINT* monitor);

引数

enc_id エンコーダの識別子
monitor モニタの状態を取得するアドレス

値	説明
TRUE	モニタ中である
FALSE	モニタ中でない

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Get_VideoCD_Mode

Video CD モードに設定されているか調べます。

書式

ENC_RETURN ENC_Get_VideoCD_Mode(UINT enc_id, LPINT lpbEnable);

引数

enc_id　　エンコーダの識別子
lpbEnable　　**Video CD** モードに設定されているかを調べるための値を取得するアドレス

値	説明
TRUE	**Video CD** モードである
FALSE	**Video CD** モードでない

戻り値

成功　　**ENC_SUCCESS**
失敗　　**ENC_FAIL**

参照

ENC_Set_VideoCD_Mode

ENC_Set_VideoCD_Mode

Video CD モードに設定します。

書式

ENC_RETURN ENC_Set_VideoCD_Mode(UINT enc_id, INT bEnable);

引数

enc_id	エンコーダの識別子
bEnable	**Video CD** モードを設定する

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

●Video CD モードを有効にすると、エンコード開始時に以下の値が自動的に設定されます。

項目	値
MPEG の種類	**MPEG1**
ビデオ入力フォーマット	**SIF**
ビデオ入力サイズ	**NTSC　352×240** **PAL　352×288**
ビデオビットレート	**1150000 bps**
CBR/VBR	**CBR**
オーディオビットレート	**224000 bps**
サンプリングレート	**44100 Hz**
レイヤ	レイヤⅡ
オーディオモード	モノラル以外（モノラルに設定されている場合は、ステレオに変更されます。）
エンファシス	**CCITT** 以外（**CCITT** に設定されている場合は、無しに変更されます。）
GOP 完結モード／**GOP** 非完結モード	**GOP** 非完結モード

参照

ENC_Get_VideoCD_Mode,ENC_Get_Video_Encode_Parameter, ENC_Get_Audio_Parameter, ENC_Get_Audio_Encode_Parameter

ENC_Get_BSS_Parameter

メモリ転送用パラメータを取得します。

書式

ENC_RETURN ENC_Get_BSS_Parameter(UINT enc_id,
ENC_BSS_PARAMETER* pParam);

引数

enc_id	エンコーダの識別子
pParam	メモリ転送用パラメータを取得する **ENC_BSS_PARAMETER** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Initialize_Ex

ENC_Set_BSS_Parameter

メモリ転送用パラメータを設定します。

書式

ENC_RETURN ENC_Set_BSS_Parameter(UINT enc_id,
ENC_BSS_PARAMETER* pParam);

引数

enc_id	エンコーダの識別子
pParam	メモリ転送用パラメータを設定する **ENC_BSS_PARAMETER** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Initialize_Ex

ENC_Get_BSS_Parameter_AV

エレメンタリストリームでのメモリ転送用パラメータを取得します。

書式

ENC_RETURN ENC_Get_BSS_Parameter_AV(UINT enc_id,
ENC_BSS_PARAMETER* pParamAudio,
ENC_BSS_PARAMETER* pParamVideo);

引数

enc_id	エンコーダの識別子
pParamAudio	オーディオのメモリ転送用パラメータを取得する **ENC_BSS_PARAMETER** 構造体のポインタ
pParamVideo	ビデオのメモリ転送用パラメータを取得する **ENC_BSS_PARAMETER** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

● エレメンタリストリームのメモリ転送では、**ENC_BSS_PARAMETER** 構造体の **fType** メンバは使用しません。

参照

ENC_Initialize_Ex

ENC_Set_BSS_Parameter_AV

エレメンタリストリームでのメモリ転送用パラメータを設定します。

書式

ENC_RETURN ENC_Set_BSS_Parameter_AV(UINT enc_id,
ENC_BSS_PARAMETER* pParamAudio,
ENC_BSS_PARAMETER* pParamVideo);

引数

enc_id	エンコーダの識別子
pParamAudio	オーディオのメモリ転送用パラメータを設定する **ENC_BSS_PARAMETER** 構造体のポインタ
pParamVideo	ビデオのメモリ転送用パラメータを設定する **ENC_BSS_PARAMETER** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

● エレメンタリストリームのメモリ転送では、**ENC_BSS_PARAMETER** 構造体の **fType** メンバは使用しません。

参照

ENC_Initialize_Ex

ENC_Get_Overlay_Parameter

オーバーレイ表示パラメータを取得します。

書式

ENC_RETURN ENC_Get_Overlay_Parameter(UINT enc_id,
ENC_OVERLAY_PARAMETER* pParam);

引数

enc_id	エンコーダの識別子
pParam	オーバーレイ表示パラメータを取得する **ENC_OVERLAY_PARAMETER** 構造体のポインタ

ENC_OVERLAY_PARAMETER 構造体の定義

オーバーレイ表示パラメータ情報が格納される。

```
typedef struct {
    UINT    nOverlayBrightness;
    int     nOverlayContrast;
    int     nOverlaySaturation;
} ENC_OVERLAY_PARAMETER;
```

nOverlayBrightness 明るさの値が格納される。

値	説明
ENCMIN_VIDEO_OVERLAY_BRIGHTNESS	最小値
ENCMAX_VIDEO_OVERLAY_BRIGHTNESS	最大値
ENCDEF_VIDEO_OVERLAY_BRIGHTNESS	デフォルト値

nOverlayContrast コントラストの値が格納される。

値	説明
ENCMIN_VIDEO_OVERLAY_CONTRAST	最小値
ENCMAX_VIDEO_OVERLAY_CONTRAST	最大値
ENCDEF_VIDEO_OVERLAY_CONTRAST	デフォルト値

nOverlaySaturation 色の濃さの値が格納される。

値	説明
ENCMIN_VIDEO_OVERLAY_SATURATION	最小値
ENCMAX_VIDEO_OVERLAY_SATURATION	最大値
ENCDEF_VIDEO_OVERLAY_SATURATION	デフォルト値

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Set_Overlay_Parameter

オーバーレイ表示パラメータを設定します。

書式

ENC_RETURN ENC_Set_Overlay_Parameter(UINT enc_id,
ENC_OVERLAY_PARAMETER* pParam);

引数

enc_id	エンコーダの識別子
pParam	オーバーレイ表示パラメータを設定する **ENC_OVERLAY_PARAMETER** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Get_Overlay_Parameter

ENC_Get_Video_Parameter

ビデオパラメータを取得します。

書式

ENC_RETURN ENC_Get_Video_Parameter(UINT enc_id,
ENC_VIDEO_PARAMETER* Video_Param);

引数

enc_id エンコーダの識別子
Video_Param ビデオパラメータを取得する **ENC_VIDEO_PARAMETER** 構造体のポインタ

ENC_VIDEO_PARAMETER 構造体の定義

ビデオパラメータ情報が格納される。

```
typedef struct {
	int	TV_System;
	int	fInputSource;
	UINT	nInputBrightness;
	int	nInputContrast;
	int	nInputHue;
	int	nInputSaturation;
	BOOL	fOutputMonitor;
} ENC_VIDEO_PARAMETER;
```

TV_System 放送規格の種類が格納される。

値	説明
ENC_VIDEO_TV_SYSTEM_NTSC	NTSC
ENC_VIDEO_TV_SYSTEM_PAL	PAL

fInputSource 入力ソースの種類が格納される。

値	説明
ENC_VIDEO_INPUT_SOURCE_COMPOSITE	Composite
ENC_VIDEO_INPUT_SOURCE_SVIDEO	S-Video

nInputBrightness ビデオ入力の明るさの値が格納される。

値	説明
ENCMIN_VIDEO_INPUT_BRIGHTNESS	最小値
ENCMAX_VIDEO_INPUT_BRIGHTNESS	最大値
ENCDEF_VIDEO_INPUT_BRIGHTNESS	デフォルト値

nInputContrast　　　ビデオ入力のコントラストの値が格納される。

値	説明
ENCMIN_VIDEO_INPUT_CONTRAST	最小値
ENCMAX_VIDEO_INPUT_CONTRAST	最大値
ENCDEF_VIDEO_INPUT_CONTRAST	デフォルト値

nInputHue　　　ビデオ入力の色合いの値が格納される。

値	説明
ENCMIN_VIDEO_INPUT_HUE	最小値
ENCMAX_VIDEO_INPUT_HUE	最大値
ENCDEF_VIDEO_INPUT_HUE	デフォルト値

nInputSaturation　　　ビデオ入力の色の濃さの値が格納される。

値	説明
ENCMIN_VIDEO_INPUT_SATURATION	最小値
ENCMAX_VIDEO_INPUT_SATURATION	最大値
ENCDEF_VIDEO_INPUT_SATURATION	デフォルト値

fOutputMonitor　　　ビデオ出力状態が格納される。

値	説明
TRUE	ビデオ出力 **ON**
FALSE	ビデオ出力 **OFF**

戻り値

成功　　　**ENC_SUCCESS**
失敗　　　**ENC_FAIL**

ENC_Set_Video_Parameter

ビデオ・パラメータを設定します。

書式

ENC_RETURN ENC_Set_Video_Parameter(UINT enc_id,
ENC_VIDEO_PARAMETER* Video_Param);

引数

enc_id	エンコーダの識別子
Video_Param	ビデオパラメータを設定する **ENC_VIDEO_PARAMETER** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

● モニタ中に **ENC_VIDEO_PARAMETER** 構造体の以下のメンバの値を変更する場合は、**ENC_Stop_Monitor** によりモニタを停止させてから **ENC_Set_Video_Parameter** により値を変更します。モニタを再開する場合は、この後に **ENC_Start_Monitor** を行ないます。

- **TV_System**
- **fInputSource**
- **fOutputMonitor**

● デジタル・ビデオ入力の時は、**ENC_VIDEO_PARAMETER** 構造体の以下のメンバについて設定は可能ですが、反映はされません。

- **TV_System**
- **fInputSource**
- **nInputBrightness**
- **nInputContrast**
- **nInputHue**
- **nInputHue**
- **nInputSaturation**

参照

ENC_Get_Video_Parameter, ENC_Start_Monitor , ENC_Stop_Monitor, ENC_Set_Digital_Video_Input, ENC_Get_Digital_Video_Input

ENC_Get_Video_Parameter2

ビデオパラメータ 2 を取得する。

書式

ENC_RETURN ENC_Get_Video_Parameter2(UINT enc_id,
ENC_VIDEO_PARAMETER2* Video_Param);

引数

enc_id エンコーダの識別子
Video_Param 拡張ビデオパラメータ **2** を取得する
ENC_VIDEO_PARAMETER2 構造体のポインタ

ENC_VIDEO_PARAMETER2 構造体の定義

拡張ビデオパラメータ 2 情報が格納される。

```
typedef struct {
        DWORD                       dwSize;
        ENC_VIDEO_PARAMETER         EncVideoParam;
        int                         nVideoProcessing;
} ENC_VIDEO_PARAMETER2;
```

dwSize **ENC_VIDEO_PARAMETER2**
構造体のサイズを指定してください。
dwSize = sizeof(ENC_VIDEO_PARAMETER2)
EncVideoParam ビデオパラメータ情報が格納される
nVideoProcessing ビデオプロセッシングの設定状態が格納される。

値	説明
ENC_VIDEO_PROCESSING_3DYC	**3** 次元 **YC** 分離を使用します。 コンポジット入力時のみ有効です。
ENC_VIDEO_PROCESSING_NONE	ビデオプロセッシングを使用しません
ENC_VIDEO_PROCESSING_NR_W	ノイズリダクション弱を使用します。
ENC_VIDEO_PROCESSING_NR_S	ノイズリダクション強を使用します。

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

参照

ENC_Get_Video_Parameter

補足

●**ENC_VIDEO_PARAMETER2** の **nVideoProcessing** メンバは **MVR-D2200** 専用です。
MVR-D2200 でない場合は使用しても何も反映されません。

●**ENC_VIDEO_PARAMETER2** 構造体については、今後必要に応じてパラメータが拡張される場合があります。

ENC_Set_Video_Parameter2

ビデオパラメータ 2 を設定します。

書式

ENC_RETURN ENC_Set_Video_Parameter2(UINT enc_id,
ENC_VIDEO_PARAMETER2* Video_Param);

引数

enc_id	エンコーダの識別子
Video_Param	ビデオパラメータ **2** を設定する **ENC_VIDEO_PARAMETER2** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Get_Video_Parameter2

ENC_Get_Video_Encode_Parameter

ビデオエンコード用パラメータを取得します。

書式

ENC_RETURN ENC_Get_Video_Encode_Parameter(UINT enc_id,
ENC_VIDEO_ENCODE_PARAMETER* Video_Param);

引数

enc_id エンコーダの識別子
Video_Param ビデオエンコード用パラメータを取得する**ENC_VIDEO_ENCODE_PARAMETER**構造体のポインタ

ENC_VIDEO_ENCODE_PARAMETER 構造体の定義

ビデオエンコード用パラメータ情報が格納される。

```
typedef struct {
        int             V_CodingMode;
        DWORD           fProfileAndLevel;
        DWORD           fConvertType;
        WORD            wImageLeft;
        WORD            wImageTop;
        WORD            wImageWidth;
        WORD            wImageHeight;
        DWORD           dwBitrate;
        int             EncodeType;
        int             GopPattern;
} ENC_VIDEO_ENCODE_PARAMETER;
```

V_CodingMode **MPEG** の種類が格納される。

値	説明
ENCVEPARAM_V_CODING_MODE_MPEG1	MPEG1
ENCVEPARAM_V_CODING_MODE_MPEG2	MPEG2

fProfileAndLevel プロファイルとレベルの種類が格納される。
MPEG1 では使用しません。

値	説明
ENCVEPARAM_PROFILE_AND_LEVEL_MP_ML	MP@ML
ENCVEPARAM_PROFILE_AND_LEVEL_MP_LL	MP@LL
ENCVEPARAM_PROFILE_AND_LEVEL_SP_ML	SP@ML

fConvertType ビデオ入力フォーマットの種類が格納される。
MPEG1 の場合は、**SIF** 固定になります。

値	説明
ENCVEPARAM_CONVERT_TYPE_STANDARD	**Standard**（幅 **1**×高さ **1**）
ENCVEPARAM_CONVERT_TYPE_SIF	**SIF**（幅 **1/2**×高さ **1/2**）
ENCVEPARAM_CONVERT_TYPE_HALF_D1	**Half-D1**（幅 **1/2**×高さ **1**）

wImageLeft 予約 **0** を渡す。
wImageTop 予約 **0** を渡す。
wImageWidth ビデオ入力サイズ（幅）が格納される。
設定時は、**16** 画素の整数倍を指定する。
wImageHeight ビデオ入力サイズ（高さ）が格納される。
設定時は、**16** 画素の整数倍を指定する。

MPEG の種類	プロファイルとレベル	ビデオ入力フォーマット	ビデオ入力サイズ	
			NTSC	**PAL**
MPEG1	設定不可	**SIF**	**352**×**240**	**352**×**288**
MPEG2	**MP@ML**	**Standard**	**720**×**480**	**720**×**576**
		SIF	**352**×**240**	**352**×**288**
		Half-D1	**352**×**480**	**352**×**576**
	MP@LL	**SIF**	**352**×**240**	**352**×**288**
	SP@ML	**Standard**	**720**×**480**	**720**×**576**
		SIF	**352**×**240**	**352**×**288**
		Half-D1	**352**×**480**	**352**×**576**

dwBitrate 平均ビデオビットレートの値が格納される。（単位：**bps**）
設定時は、**400bps** の整数倍を指定する。

MPEG の種類	プロファイルとレベル	ビデオ入力フォーマット	平均ビデオビットレートの設定範囲
MPEG1	設定不可	**SIF**	**1,000,000**～**1,800,000 bps**
MPEG1(Video CD)	設定不可	**SIF**	**1,150,000 bps** のみ
MPEG2	**MP@ML**	**Standard**	**4,000,000**～**15,000,000 bps**
		SIF	**2,000,000**～**8,000,000 bps**
		Half-D1	**2,000,000**～**8,000,000 bps**
	MP@LL	**SIF**	**2,000,000**～**4,000,000 bps**
	SP@ML	**Standard**	**4,000,000**～**15,000,000 bps**
		SIF	**2,000,000**～**8,000,000 bps**
		Half-D1	**2,000,000**～**8,000,000 bps**

EncodeType **CBR/VBR** の選択状態が格納される。

値	説明
ENCVEPARAM_ENCODE_TYPE_CBR	**CBR**
ENCVEPARAM_ENCODE_TYPE_VBR	**VBR**

GopPattern **GOP** パターンの選択状態が格納される。

値	説明
ENCVEPARAM_GOP_PATTERN_IFRAME	**I Frame**
ENCVEPARAM_GOP_PATTERN_IBBP	**IBBP**

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Set_Video_Encode_Parameter

ビデオエンコード用パラメータを設定します。

書式

ENC_RETURN ENC_Set_Video_Encode_Parameter(UINT enc_id,
ENC_VIDEO_ENCODE_PARAMETER* Video_Param);

引数

enc_id エンコーダの識別子
Video_Param ビデオエンコード用パラメータを設定する **ENC_VIDEO_ENCODE_PARAMETER** 構造体のポインタ

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

補足

●**ENC_Set_Video_Encode_Parameter** を実行すると、以下の値が自動的に設定されます。

設定項目	値		
	I Frame	IBBP	
		SP@ML 以外	SP@ML
GOP 内のピクチャ枚数	1	15	15
GOP 内の I ピクチャまたは P ピクチャが現れる周期	1	3	1
VBR での最大ビデオビットレート	ビデオビットレートの 2 倍	現在の設定	
ビデオエレメンタリストリームに対する Closed GOP の設定	OFF		
アスペクト比	4:3		

参照

ENC_Get_Video_Encode_Parameter, ENC_Get_Video_Encode_Parameter_Ex

ENC_Get_Video_Encode_Parameter_Ex

拡張ビデオエンコード用パラメータを取得します。

書式

ENC_RETURN ENC_Get_Video_Encode_Parameter_Ex(UINT enc_id,
ENC_VIDEO_ENCODE_PARAMETER_EX* Video_Param);

引数

enc_id エンコーダの識別子
Video_Param 拡張ビデオエンコード用パラメータを取得する **ENC_VIDEO_ENCODE_PARAMETER_EX** 構造体のポインタ

ENC_VIDEO_ENCODE_PARAMETER_EX 構造体の定義

拡張ビデオエンコード用パラメータ情報が格納される。

```
typedef struct {
    ENC_VIDEO_ENCODE_PARAMETER      EncodeParam;
    WORD                            cPictureInterval;
    WORD                            cPicturePerGop;
    WORD                            dwVbrPeak;
    BOOL                            fClosedGop;
    BOOL                            fGopComplete;
    WORD                            wAspectRatio;
    BOOL                            fLowDelay;
} ENC_VIDEO_ENCODE_PARAMETER_EX;
```

EncodeParam ビデオエンコード用パラメータ情報が格納される。
cPictureInterval GOP 内の I ピクチャまたは P ピクチャが現れる周期が格納される。

値	説明
ENCVEPARAMEX_MIN_PICTURE_INTERVAL	最小値
ENCVEPARAMEX_MAX_PICTURE_INTERVAL	最大値

cPicturePerGop GOP 内のピクチャ枚数が格納される。

値	説明
ENCVEPARAMEX_MIN_PICTURE_PER_GOP	最小値
ENCVEPARAMEX_MAX_PICTURE_PER_GOP	最大値

dwVbrPeak　　　　**VBR** での最大ビデオビットレートが格納される。（単位：**bps**）

MPEG の種類	最大ビデオビットレートの設定範囲
MPEG1	平均ビデオビットレートの設定値～**1,800,000 bps**
MPEG1 (Video CD)	設定不可
MPEG2 (MP@ML)	平均ビデオビットレートの設定値～**15,000,000 bps**
MPEG2 (MP@LL)	平均ビデオビットレートの設定値～**4,000,000 bps**
MPEG2 (SP@ML)	平均ビデオビットレートの設定値～**15,000,000 bps**

fClosedGop　　　　ビデオエレメンタリストリームに対する **Closed GOP** の設定状態が格納される。

値	説明
TRUE	**ON**
FALSE	**OFF**

fGopComplete　　　　**GOP** 完結プログラムストリームの作成状態が格納される。

値	説明
TRUE	**GOP** 完結モード
FALSE	**GOP** 非完結モード

wAspectRatio　　　　アスペクト比の値が格納される。

値	説明
ENCVEPARAMEX_ASPECT_RATIO_4_3	**4:3**
ENCVEPARAMEX_ASPECT_RATIO_16_9	**16:9**

fLowDelay　　　　**Low Delay** の設定状態が格納される。

値	説明
TRUE	**ON**
FALSE	**OFF**

戻り値

成功　　　　**ENC_SUCCESS**
失敗　　　　**ENC_FAIL**

補足

● **ENC_VIDEO_ENCODE_PARAMETER_EX** 構造体については、今後必要に応じてパラメータが拡張される場合があります。

参照

ENC_Get_Video_Encode_Parameter

ENC_Set_Video_Encode_Parameter_Ex

拡張ビデオエンコード用パラメータを設定します。

書式

ENC_RETURN ENC_Set_Video_Encode_Parameter_Ex(UINT enc_id,
ENC_VIDEO_ENCODE_PARAMETER_EX* Video_Param);

引数

enc_id エンコーダの識別子
Video_Param 拡張ビデオエンコード用パラメータを設定する **ENC_VIDEO_ENCODE_PARAMETER_EX** 構造体のポインタ

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

補足

- **ENC_VIDEO_ENCODE_PARAMETER_EX** 構造体の **fGopComplete** により **GOP** 完結プログラムストリームを作成する設定を行なった場合、以下の項目に固定値が設定されます。

設定項目	値
レイヤ	レイヤⅡ
サンプリング周波数	**48kHz**

- **ENC_VIDEO_ENCODE_PARAMETER_EX** 構造体の **wAspectRatio** により **TRUE** を設定した場合、表示領域サイズの情報がビデオエレメンタリストリーム内に付加されます。この情報は **MPEG1** では付加されません。

画像サイズと表示領域サイズの関係は次のようになります。

ビデオフォーマット	画像サイズ	表示領域サイズ
NTSC Standard	**720×480**	**540×480**
NTSC SIF	**352×240**	**264×240**
NTSC Half-D1	**352×480**	**264×480**
PAL Standard	**720×576**	**540×576**
PAL SIF	**352×288**	**264×288**
PAL Half-D1	**352×576**	**264×576**

● **ENC_VIDEO_ENCODE_PARAMETER_EX** 構造体の **fLowDelay** により **Low Delay** を **ON** にする設定を行なった場合、同じく構造体の **cPictureInterval** は **ENCVEPARAMEX_MIN_PICTURE_INTERVAL** に設定されます。

参照

ENC_Get_Video_Encode_Parameter, ENC_Get_Video_Encode_Parameter_Ex, ENC_Get_Audio_Parameter, ENC_Get_Audio_Encode_Parameter

ENC_Get_Video_Encode_Parameter_Ex2

拡張ビデオエンコード用パラメータ 2 を取得する。

書式

ENC_RETURN ENC_Get_Video_Encode_Parameter_Ex2(UINT enc_id,
ENC_VIDEO_ENCODE_PARAMETER_EX2* Video_Param);

引数

enc_id エンコーダの識別子
Video_Param 拡張ビデオエンコード用パラメータ **2** を取得する
ENC_VIDEO_ENCODE_PARAMETER_EX2 構造体のポインタ

ENC_VIDEO_ENCODE_PARAMETER_EX2 構造体の定義

拡張ビデオエンコード用パラメータ 2 情報が格納される。

```
typedef struct {
    DWORD                              dwSize;
    ENC_VIDEO_ENCODE_PARAMETER_EX      EncodeParamEx;
    BOOL                               fBottomFieldFirst;
} ENC_VIDEO_ENCODE_PARAMETER_EX2;
```

dwSize **ENC_VIDEO_ENCODE_PARAMETER_EX2**
構造体のサイズを指定してください。
dwSize = sizeof(ENC_VIDEO_ENCODE_PARAMETER_EX2)
EncodeParamEx 拡張ビデオエンコード用パラメータ情報が格納される
fBottomFieldFirst フィールドオーダーの設定状態が格納される。

値	説明
TRUE	ボトムファースト
FALSE	トップファースト

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

参照

ENC_Get_Video_Encode_Parameter_Ex

補足

●**ENC_VIDEO_ENCODE_PARAMETER_EX2** 構造体については、今後必要に応じてパラメ

ータが拡張される場合があります。

ENC_Set_Video_Encode_Parameter_Ex2

拡張ビデオエンコード用パラメータ 2 を設定します。

書式

ENC_RETURN ENC_Set_Video_Encode_Parameter_Ex2(UINT enc_id,
ENC_VIDEO_ENCODE_PARAMETER_EX2* Video_Param);

引数

enc_id	エンコーダの識別子
Video_Param	拡張ビデオエンコード用パラメータ **2** を設定する **ENC_VIDEO_ENCODE_PARAMETER_EX2** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Set_Video_Encode_Parameter_Ex

ENC_Get_Audio_Format

オーディオ形式を取得します。

書式

ENC_RETURN ENC_Get_Audio_Format (UINT enc_id,
ENC_AUDIO_FORMAT *pAudioFormat);

引数

enc_id エンコーダの識別子
pAudioFormat オーディオ形式を取得するアドレス

値	説明
ENC_AUDIO_FORMAT_MPEG	**MPEG**
ENC_AUDIO_FORMAT_PCM_MONO	**PCM** モノラル
ENC_AUDIO_FORMAT_PCM_STEREO	**PCM** ステレオ

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Set_Audio_Format

オーディオ形式を設定します。

書式

ENC_RETURN ENC_Set_Audio_Format(UINT enc_id,
ENC_AUDIO_FORMAT AudioFormat);

引数

enc_id	エンコーダの識別子
AudioFormat	オーディオ形式を設定する

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Get_Audio_Format

ENC_Get_Audio_Parameter

オーディオパラメータを取得します。

書式

ENC_RETURN ENC_Get_Audio_Parameter(UINT enc_id,
ENC_AUDIO_PARAMETER* Audio_Param);

引数

enc_id エンコーダの識別子
Audio_Param オーディオパラメータを取得する **ENC_AUDIO_PARAMETER** 構造体のポインタ

ENC_AUDIO_PARAMETER 構造体の定義

オーディオパラメータ情報が格納される。

```
typedef struct {
        DWORD      dwSamplingFrequency;
        BOOL       fOutputMonitor;
        BOOL       fOutputMute;
        DWORD      dwLeftOutputAtt;
        DWORD      dwRightOutputAtt;
        int        nLeftInputGain;
        int        nRightInputGain;
} ENC_AUDIO_PARAMETER;
```

dwSamplingFrequency サンプリング周波数が格納される。(単位:**Hz**)
設定値:**32000/44100/48000**

fOutputMonitor オーディオ出力状態が格納される。

値	説明
TRUE	オーディオ出力 **ON**
FALSE	オーディオ出力 **OFF**

fOutputMute ミュート状態が格納される。

値	説明
TRUE	ミュート **ON**
FALSE	ミュート **OFF**

dwLeftOutputAtt　　左の音量が格納される。

値	説明
ENCMIN_AUDIO_OUTPUT_ATT	最小値
ENCMAX_AUDIO_OUTPUT_ATT	最大値

dwRightOutputAtt　　右の音量が格納される。（値は **dwLeftOutputAtt** と同じ）

nLeftInputGain　　左のオーディオ **PCM** 入力に対する **GAIN** が格納される。

値	説明
ENCMIN_AUDIO_INPUT_GAIN	最小値
ENCMAX_AUDIO_INPUT_GAIN	最大値

nRightInputGain　　右のオーディオ **PCM** 入力に対する **GAIN** が格納される。
（値は **nLeftInputGain** と同じ）

戻り値

成功　　**ENC_SUCCESS**
失敗　　**ENC_FAIL**

ENC_Set_Audio_Parameter

オーディオパラメータを設定します。

書式

ENC_RETURN ENC_Set_Audio_Parameter(UINT enc_id,
ENC_AUDIO_PARAMETER* Audio_Param);

引数

enc_id	エンコーダの識別子
Audio_Param	オーディオパラメータを設定する **ENC_AUDIO_PARAMETER** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

- モニタ中に **ENC_AUDIO_PARAMETER** 構造体の **dwSamplingFrequency**、**fOutputMonitor** の値を変更する場合は、**ENC_Stop_Monitor** によりモニタを停止させてから **ENC_Set_Audio_Parameter** により値を変更します。モニタを再開する場合は、この後に **ENC_Start_Monitor** を行ないます。

参照

ENC_Get_Audio_Parameter, ENC_Stop_Monitor, ENC_Start_Monitor

ENC_Get_Audio_Encode_Parameter

オーディオエンコード用パラメータを取得します。

書式

ENC_RETURN ENC_Get_Audio_Encode_Parameter(UINT enc_id,
ENC_AUDIO_ENCODE_PARAMETER* Audio_Param);

引数

enc_id エンコーダの識別子
Audio_Param オーディオエンコード用パラメータを取得する **ENC_AUDIO_ENCODE_PARAMETER** 構造体のポインタ

ENC_AUDIO_ENCODE_PARAMETER 構造体の定義

オーディオパラメータ情報が格納される。

```
typedef struct {
        DWORD       nLayer;
        BOOL        fProtection;
        int         AudioBitrate;
        int         Channel_Num;
        DWORD       fModeEx;
        BOOL        fCopyright;
        BOOL        fOriginal;
        BOOL        fEmphasis;
} ENC_AUDIO_ENCODE_PARAMETER;
```

nLayer 予約 **ENCAEPARAM_LAYER2** を渡す。

fProtection プロテクションの状態が格納される。

値	説明
TRUE	エラーチェック有り
FALSE	エラーチェック無し

AudioBitrate　　オーディオビットレートが格納される。

Channel_Num　　プロテクションの状態が格納される。

値	説明
ENCAEPARAM_CHANNEL_NUM_STEREO	ステレオ
ENCAEPARAM_CHANNEL_NUM_MONO	モノラル
ENCAEPARAM_CHANNEL_NUM_DUAL_MONO	デュアル・チャネル
ENCAEPARAM_CHANNEL_NUM_JOINT_STEREO	ジョイント・ステレオ

fModeEx　　予約 **0** を渡す。

fCopyright　　コピーライトの状態が格納される。

値	説明
TRUE	著作権有り
FALSE	著作権無し

fOriginal　　オリジナル／コピーの状態が格納される。

値	説明
TRUE	オリジナル
FALSE	コピー

fEmphasis　　エンファシス特性の状態が格納される。

値	説明
ENCAEPARAM_EMPHASIS_NONE	無し
ENCAEPARAM_EMPHASIS_5015	**50/15 μ s**
ENCAEPARAM_EMPHASIS_CCITT	**CCITT J.17**

戻り値

成功　　**ENC_SUCCESS**
失敗　　**ENC_FAIL**

ENC_Set_Audio_Encode_Parameter

オーディオエンコード用パラメータを設定します。

書式

ENC_RETURN ENC_Set_Audio_Encode_Parameter(UINT enc_id,
ENC_AUDIO_ENCODE_PARAMETER* Audio_Param);

引数

enc_id	エンコーダの識別子
Audio_Param	オーディオエンコード用パラメータを設定する **ENC_AUDIO_ENCODE_PARAMETER** 構造体のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Get_Audio_Encode_Parameter

ENC_Init_Movie

エンコードを開始待ち状態にします。

書式

ENC_RETURN ENC_Init_Movie(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Record_Movie

エンコードを開始します。

書式

ENC_RETURN ENC_Record_Movie(UINT enc_id, UINT enc_mode_flag);

引数

enc_id エンコーダの識別子
enc_mode_flag エンコードを行なう種類を設定する

値	説明
ENC_RECORD_AUDIO	オーディオのみ
ENC_RECORD_VIDEO	ビデオのみ（エレメンタリストリーム）
ENC_RECORD_AUDIO | ENC_RECORD_VIDEO	オーディオ＋ビデオ
ENC_RECORD_PS	**MPEG1**：システムストリーム **MPEG2**：プログラムストリーム

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

補足

- メモリ転送を行なう設定で **ENC_Initialize** を実行した場合は、次の条件でのエンコードは行なえません。

①オーディオ＋ビデオのエンコード

②オーディオ形式を **PCM** に設定した場合のオーディオのみのエンコード

参照

ENC_Initialize, ENC_Initialize_Ex, ENC_Get_Audio_Format, ENC_Set_Audio_Format

ENC_Stop

エンコードを停止します。

書式

ENC_RETURN ENC_Stop(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Get_BitmapBits

静止画（BMP）をバッファで取得します。

書式

ENC_RETURN ENC_Get_BitmapBits(UINT enc_id, LPBYTE* lpByte, LPDWORD pdwLen, MVRSTILLCAPPARAM pParam);

引数

enc_id	エンコーダの識別子
lpByte	データを格納するバッファへのポインタを格納するポインタ
pdwLen	取得した静止画のバイト数を格納するポインタ
pParam	**MVRSTILLCAPPARAM** 構造体

MVRSTILLCAPPARAM 構造体の定義

```
typedef struct {
        DWORD                   dwSize;
        INT                     nFrameCap;
        BOOL                    fExpandRGB;
        double                  dCorrectGamma;
        MVR_UYVY2RGB_FUNC       pfnuyvy2rgb;
} MVRSTILLCAPPARAM;
```

dwSize この構造体のサイズを指定します。
dwSize = sizeof(MVRSTILLCAPPARAM)

nFrameCap フレームのキャプチャ方法を指定します。

値	説明
MVRSTILL_FRAME	1フレームをキャプチャします。
MVRSTILL_ODD_FIELD	奇数ラインのみ取得し2倍に引き伸ばします。
MVRSTILL_EVEN_FIELD	偶数ラインのみ取得し2倍に引き伸ばします。

fExpandRGB **TRUE** の場合、**YUV → RGB** 変換時に、**[235-16]** の範囲を**[0-255]**の範囲へ拡大する変換を行います。

dCorrectGamma ガンマ補正値を設定します。標準で NTSC の時 2.2 で PAL の時 2.8 です。使用しないときは1に設定します。

pfnuyvy2rgb 変換を行う関数を指定できます。指定しない場合は NULL でないといけません。

MVR_UYVY2RGB_FUNC 関数の定義

YUV -> RGB に変換するための関数

INT ConvertYUVtoRGB(BYTE u1, BYTE y1, BYTE v1, BYTE y2, COLORREF *pColor1, COLORREF *pColor2)

u1	変換元輝度信号と赤色成分の差
y1	**RGB 1** バイト目用の 変換元 輝度信号
v1	変換元 輝度信号と青色成分の差
y2	**RGB 2** バイト目用の 変換元 輝度信号
***pColor1**	変換後 **RGB 1** バイト目
***pColor2**	変換後 **RGB 2** バイト目

戻り値

ConvertYUVtoRGB 関数は必ず 1 を返してください。

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

- エンコード中でもキャプチャ可能です。
- キャプチャ時に映像の DMA の転送を 1 度止めるので PC 上のオーバーレイの映像が一瞬停止します。(外部モニタへ出力されている映像は停止しません)
- 取得できるサイズはソースサイズです。(NTSC : 720 × 480、PAL : 720 × 576)

ENC_Save_DIB

静止画(BMP)をファイルに保存します。

書式

ENC_RETURN ENC_Save_DIB(UINT enc_id, LPCTSTR lpszFileName, MVRSTILLCAPPARAM pParam);

引数

enc_id	エンコーダの識別子
lpszFileName	保存するファイル名
pParam	静止画キャプチャパラメータを設定する **MVRSTILLCAPPARAM** 構造体

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Get_BitmapBits の補足を確認してください。

ENC_Save_JPEG

静止画(JPEG)をファイルに保存します。

書式

ENC_RETURN ENC_Save_JPEG(UINT enc_id, LPCTSTR lpszFileName, UINT nQuality, MVRSTILLCAPPARAM pParam);

引数

enc_id	エンコーダの識別子
lpszFileName	保存するファイル名
nQuality	**JPEG** の圧縮率(**0**～**9** の範囲で指定できます)を指定します。 **0 :** 約 **1/5** **1 :** 約 **1/7** **2 :** 約 **1/10** **3 :** 約 **1/12** **4 :** 約 **1/15** **5 :** 約 **1/17** **6 :** 約 **1/20** **7 :** 約 **1/25** **8 :** 約 **1/30** **9 :** 約 **1/40**
pParam	静止画キャプチャパラメータを設定する **MVRSTILLCAPPARAM** 構造体

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Get_BitmapBits の補足を確認してください

ENC_Get_Record_Time

エンコードを行なう時間を取得します。

書式

ENC_RETURN ENC_Get_Record_Time(UINT enc_id, UINT* rec_time, UINT* enabled);

引数

enc_id	エンコーダの識別子
rec_time	エンコードを行なう時間を取得するアドレス
enabled	エンコードを行なう時間制限の状態を取得するアドレス

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Set_Record_Time

ENC_Set_Record_Time

エンコードを行なう時間を設定します。

書式

ENC_RETURN ENC_Set_Record_Time(UINT enc_id, UINT rec_time, UINT enabled);

引数

enc_id エンコーダの識別子
rec_time エンコードを行なう時間（単位：秒）
enabled エンコードの時間制限

値	説明
TRUE	時間制限有り
FALSE	時間制限なし

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Get_Movie_File

エンコードを行なうファイル名を取得します。

書式

ENC_RETURN ENC_Get_Movie_File(UINT enc_id, UINT enc_mode_flag, LPTSTR enc_file);

引数

enc_id	エンコーダの識別子
enc_mode_flag	エンコードを行なうファイルの種類
enc_file	エンコードを行なうファイル名を取得するアドレス

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

参照

ENC_Set_Movie_File

ENC_Set_Movie_File

エンコードを行なうファイル名を設定します。

書式

ENC_RETURN ENC_Set_Movie_File(UINT enc_id, UINT enc_mode_flag,
LPCTSTR enc_file);

引数

enc_id エンコーダの識別子
enc_mode_flag エンコードを行なうファイルの種類

値	説明
ENC_RECORD_AUDIO	オーディオファイル
ENC_RECORD_VIDEO	ビデオ(エレメンタリストリーム)ファイル
ENC_RECORD_PS	プログラムストリームファイルまたは システムストリームファイル

enc_file エンコードを行なうファイル名を設定するアドレス

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Get_Frame_Count

エンコードを行なったフレーム数を取得します。

書式

ENC_RETURN ENC_Get_Frame_Count(UINT enc_id, UINT* frame_count);

引数

enc_id	エンコーダの識別子
frame_count	フレーム数を取得するアドレス

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Get_Time

エンコードを行なった時間を取得します。

書式

ENC_RETURN ENC_Get_Time(UINT enc_id, double* time);

引数

enc_id	エンコーダの識別子
time	時間を取得するアドレス（単位：秒）

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

ENC_Detect_Video_Input_Source

ビデオ入力ソースを自動検出します。

書式

ENC_RETURN ENC_Detect_Video_Input_Source(UINT enc_id, LPINT lpfInputSource);

引数

enc_id エンコーダの識別子
lpfInputSource 検出されたビデオ入力ソースの値を取得するアドレス

値	説明
ENC_VIDEO_INPUT_SOURCE_COMPOSITE	**Composite**
ENC_VIDEO_INPUT_SOURCE_SVIDEO	**S-Video**

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

補足

- デジタル・ビデオ入力の場合は、戻り値として **ENC_FAIL** を返します。**ENC_Get_Last_Error** では、**MVR_ERROR_NO_VIDEO_SOURCE** が返ります。

参照

ENC_Get_Video_Parameter, ENC_Set_Video_Parameter, ENC_Set_Digital_Video_Input, ENC_Get_Digital_Video_Input

ENC_Set_Digital_Video_Input

アナログ･ビデオ入力とデジタル･ビデオ入力を切り換えます。

書式

ENC_RETURN ENC_Set_Digital_Video_Input(UINT enc_id, BOOL fDigital);

引数

enc_id エンコーダの識別子
fDigital ビデオ入力を示す値

値	説明
TRUE	デジタル･ビデオ入力
FALSE	アナログ･ビデオ入力

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

補足

- モニタ中にビデオ入力を変更する場合は、**ENC_Stop_Monitor** によりモニタを停止させてから **ENC_Set_Digital_Video_Input** により変更します。モニタを再開する場合は、この後に **ENC_Start_Monitor** を行ないます。

ENC_Get_Digital_Video_Input

現在のビデオ入力の状態を取得します。

書式

ENC_RETURN ENC_Get_Digital_Video_Input(UINT enc_id, LPINT lpfDigital);

引数

enc_id エンコーダの識別子
lpfDigital ビデオ入力を示す値を取得する **INT** のポインタ

値	説明
TRUE	デジタル・ビデオ入力
FALSE	アナログ・ビデオ入力

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Set_Video_Encode_File

エンコード開始時に詳細ビデオ・エンコード・パラメータを設定するためのファイル名を指定します。

書式

ENC_RETURN ENC_Set_Video_Encode_File(UINT enc_id, LPCTSTR lpszVEFile);

引数

enc_id	エンコーダの識別子
lpszVEFile	詳細ビデオ・エンコード・パラメータ用ファイル名のポインタ

戻り値

成功	**ENC_SUCCESS**
失敗	**ENC_FAIL**

補足

- **lpszVEFile** に **NULL** を設定すると、詳細ビデオ・エンコード・パラメータの設定を行いません。ただし、一度エンコードを行った後で **lpszVEFile** に **NULL** を設定しても前回設定した値がそのまま有効になります。設定を解除するには、**ENC_Terminate** によりエンコーダを終了してください。
- 詳細ビデオ・エンコード・パラメータを設定するためのファイルの内容については、付属の **VideoEncoderParams.txt** をご覧ください。

ENC_Get_IRE_Setting

NTSC セットアップレベル（黒レベル）の設定を取得します。

書式

ENC_RETURN ENC_Get_IRE_Setting(UINT enc_id, LPINT lpnIreSetting);

引数

enc_id エンコーダの識別子
lpnIreSetting セットアップレベルの情報を格納するための **INT** 変数へのポインタ

値	説明
MVR_IRE_LEVEL_0_NTSC	**0 IRE**
MVR_IRE_LEVEL_75_NTSC	**7.5 IRE**

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

ENC_Set_IRE_Setting

NTSC セットアップレベル（黒レベル）を設定します。

書式

ENC_RETURN ENC_Set_IRE_Setting(UINT enc_id, INT nIreSetting);

引数

enc_id エンコーダの識別子
nIreSetting 設定するセットアップレベル

値	説明
MVR_IRE_LEVEL_0_NTSC	**0 IRE**
MVR_IRE_LEVEL_75_NTSC	**7.5 IRE**

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

補足

- 日本では 0 IRE が標準です。
- ビデオ入力部のセットアップレベルを変更した場合 ENC_Set_Video_Parameter で設定されるビデオ入力パラメータ（明るさ、コントラスト、色合い、色の濃さ）は対応するデフォルト値に変更されます。

ENC_Get_Last_Error

最新のエラー情報を取得します。

書式

ULONG ENC_Get_Last_Error(UINT enc_id);

引数

enc_id	エンコーダの識別子

戻り値

ULONG	エラーコード（**APPENDIX** 1. エラーコード表参照）

補足

- **ENC_Get_Last_Error** を実行すると、エラー情報はクリアされます。エラーが発生していない場合や新たにエラーが発生するまでは、**MVR_ERROR_SUCCESS** が返されます。

CHAPTER 4 Video Decoder Functions

DEC_Set_Callback

コールバック関数の設定を行ないます。

書式

DEC_RETURN DEC_Set_Callback(UINT dec_id, DEC_CB_STATUS status,
DEC_CB_ERROR error);

引数

dec_id	デコーダの識別子
status	内部状態通知用コールバック関数を設定する
error	エラー通知用コールバック関数を設定する

DEC_CB_STATUS コールバック関数の定義

デコーダの内部状態を取得するためのコールバック関数

VOID CALLBACK StatusProc(UINT dec_id, DEC_STATUS_NOTIFY status);

dec_id	デコーダの識別子
status	内部状態

値	説明
DEC_NOTIFY_INITIALIZE	デコーダの初期化が行われた
DEC_NOTIFY_TERMINATE	デコーダが終了した
DEC_NOTIFY_STOP	デコードが停止した
DEC_NOTIFY_PAUSE	デコードを一時停止した
DEC_NOTIFY_PLAY	デコードを開始した

DEC_CB_ERROR コールバック関数の定義

デコード中にエラーが発生した場合に、そのエラー情報を取得するためのコールバック関数

VOID CALLBACK ErrorProc(UINT dec_id, ULONG error);

dec_id	デコーダの識別子
error	エラーコード（**APPENDIX** 1. エラーコード表参照）

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

● **DEC_CB_ERROR** で定義したコールバック関数は、エンコード中にエラーが発生した場合に実行されます。関数を実行して **DEC_FAIL** が戻り値をして返された場合は、**DEC_Get_Last_Error** によりエラー情報を取得してください。

参照

DEC_Get_Last_Error

DEC_Can_Initialize

デコーダを初期化できるか調べます。

書式

ENC_RETURN DEC_Can_Initialize(UINT enc_id);

引数

dec_id	デコーダの識別子

戻り値

デコーダの初期化が可能な場合	**DEC_SUCCESS**
すでに使用しているかデコーダを初期化できない場合	**DEC_FAIL**

DEC_Get_Codec_Config

カードに関する情報を取得します。

書式

DEC_RETURN DEC_Get_Codec_Config (UINT dec_id, DEC_CONFIG *pConfig);

引数

dec_id	デコーダの識別子
pConfig	情報を取得する **DEC_CONFIG** 構造体へのポインタ

DEC_CONFIG 構造体の定義

カードに関する情報を取得する。

```
typedef struct {
    UINT    NumberCodecs;
    UINT    CodecNumber;
    UINT    CodecCaps;
    UINT    CurrentUtilization;
} DEC_CONFIG;
```

NumberCodecs 装着されているカード枚数が格納される。
CodecNumber 情報を取得したいカード番号(**1**～**NumberCodecs**)を指定する。
CodecCaps **CodecNumber** で指定したカードに関する詳細情報フラグが格納される。

値	意味
DECCAP_NONE	情報がないことを示すフラグ
DECCAP_ENCODE	エンコードが可能であることを示すフラグ
DECCAP_DECODE	デコードが可能であることを示すフラグ
DECCAP_MONITOR	モニタが可能であることを示すフラグ
DECCAP_AVI2MPEG	**AVI** ファイルを **MPEG** に変換可能であることを示すフラグ
DECCAP_D2200	**MVR-D2200** であることを示すフラグ
DECCAP_D2200V	**MVR-D2200V** であることを示すフラグ

CurrentUtilization **CodecNumber** で指定したカードが使用状況が格納される。

値	意味
TRUE	使用している
FALSE	使用していない

戻り値

成功 **DEC_SUCCESS**

失敗　　　　　　**DEC_FAIL**

補足

- **DEC_CONFIG** 構造体の **CodecNumber** に **0** を指定すると、装着されているカード枚数と現在使用しているカード番号を取得できます。カード枚数は **NumberCodecs** に格納されます。カード番号は **CodecNumber** に格納されます。カードが使用されていない場合は **CodecNumber** に **0** が格納されます。

DEC_Set_Codec_Number

使用するカード番号を設定します。

書式

DEC_RETURN DEC_Set_Codec_Number(UINT dec_id, UINT nNumber);

引数

dec_id	デコーダの識別子
nNumber	使用するカード番号

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- **DEC_Initialize** または **DEC_Initialize_Ex** による初期化を行う前に **DEC_Set_Codec_Number** を実行してください。初期化を行う前に **DEC_Set_Codec_Number** を実行しない場合は、自動的に使用可能なカードを探して初期化を行います。
- 指定できるカード番号は、**1** から **DEC_Get_Codec_Config** で取得した **DEC_CONFIG** 構造体の **NumberCodecs** までの値です。
- カード番号に **0** を指定した場合は、初期化時に自動的に使用可能なカードを探します。

参照

DEC_Get_Codec_Config, DEC_Initialize, DEC_Initialize_Ex

DEC_Initialize

デコーダの初期化を行ないます。

書式

DEC_RETURN DEC_Initialize(UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Initialize_Ex

デコーダの初期化を行なう。

書式

DEC_RETURN DEC_Initialize_Ex(UINT dec_id, DEC_MEDIA media,
DEC_BSR_PARAMETER* pParam);

引数

dec_id デコーダの識別子
media デコードの種類

値	意味
DEC_MEDIA_FILE	ファイル
DEC_MEDIA_MEMORY	メモリ転送

pParam メモリ転送用パラメータを設定する **DEC_BSR_PARAMETER** 構造体のポインタ
pParam は、**media** が **DEC_MEDIA_MEMORY** の場合に設定する。
それ以外の場合は、**NULL** を指定する。

DEC_BSR_PARAMETER 構造体の定義

メモリ転送用パラメータ情報を設定する。

```
typedef struct {
    DEC_BSR_TYPE    fType;
    DEC_CB_BSR      pfnCallback;
    DWORD           cbBuff;
    DWORD           cBuff;
    DWORD           dwData;
} DEC_BSR_PARAMETER;
```

fType **MPEG** データの種類を設定する

値	意味
DEC_BSR_TYPE_UNKOWN	なし
DEC_BSR_TYPE_PS	**MPEG2** プログラムストリーム **MPEG1** システムストリーム
DEC_BSR_TYPE_AUDIO	**MPEG** オーディオエレメンタリストリーム
DEC_BSR_TYPE_VIDEO	**MPEG** ビデオエレメンタリストリーム

pfnCallback	メモリ転送用コールバック関数を設定する
cbBuff	コールバック時にメモリ転送するバッファサイズ サイズは、**1024byte** の倍数を指定する。
cBuff	**cbBuff** で指定したバッファの数
dwData	コールバック時に送られる **32bit** データ

DEC_CB_BSR コールバック関数の定義

デコードデータを転送するためのコールバック関数

VOID CALLBACK BsrProc(LPBYTE pbBuff, DWORD cbWrite,

LPDWORD pcbWritten, DWORD dwData);

pbBuff	デコードデータを格納するバッファのポインタ
cbWrite	デコードデータを格納するサイズ
pcbWritten	格納されたデコードデータのサイズを設定するバッファのポインタ
dwData	**DEC_BSR_PARAMETER** 構造体で設定された **dwData** の値

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Terminate

デコーダを終了します。

書式

DEC_RETURN DEC_Terminate(UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

●**DEC_Terminate** 実行後、**DEC_Set_Callback** によるコールバックの設定は全て解除されます。

参照

DEC_Set_Callback

DEC_Get_Media

デコードの種類を取得します。

書式

DEC_RETURN DEC_Get_Media(UINT dec_id, DEC_MEDIA* media);

引数

dec_id	デコーダの識別子
media	デコードの種類を取得する **DEC_MEDIA** へのポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Initialize_Ex

DEC_Set_Media

デコードの種類を設定します。

書式

DEC_RETURN DEC_Set_Media(UINT dec_id, DEC_MEDIA media);

引数

dec_id	デコーダの識別子
media	デコードの種類

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Initialize_Ex

DEC_Can_Playback

再生が可能か判断します。

書式

DEC_RETURN DEC_Can_Playback(UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

再生可能	**DEC_SUCCESS**
再生不可能	**DEC_FAIL**

DEC_Get_Status

デコーダの内部状態を取得します。

書式

DEC_RETURN DEC_Get_Status(UINT dec_id, DEC_STATUS* status);

引数

dec_id デコーダの識別子
status デコーダの内部状態を取得する **DEC_STATUS** へのポインタ

値	意味
DEC_STATUS_WAITINITIALIZE	初期化待ち状態
DEC_STATUS_WAITTERMINATE	終了待ち状態
DEC_STATUS_STOP	停止中
DEC_STATUS_PAUSE	一時停止中
DEC_STATUS_PLAY	デコード中

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

DEC_Can_Overlay_Window

オーバーレイウィンドウを使用できるかどうか判断します。

書式

DEC_RETURN DEC_Can_Overlay_Window(UINT dec_id, HANDLE hMonitor,
UINT nWidth, UINT nHeight);

引数

dec_id	デコーダの識別子
hMonitor	モニタハンドル
nWidth	オーバーレイウィンドウの幅
nHeight	オーバーレイウィンドウの高さ

戻り値

オーバーレイウィンドウが使用できる場合	**DEC_SUCCESS**
オーバーレイウィンドウが使用できない場合	**DEC_FAIL**

補足

- **hMonitor** には、マルチモニタでのプライマリまたはセカンダリのモニタハンドルを設定します。マルチモニタでない場合は、**hMonitor** に **NULL** を設定します。

DEC_Create_Overlay_Window

オーバーレイウィンドウを生成します。

書式

DEC_RETURN DEC_Create_Overlay_Window(UINT dec_id, HWND hWndParent, HWND* hWndOverlay, int x, int y, UINT nWidth, UINT nHeight);

引数

dec_id	デコーダの識別子
hWndParent	オーバーレイウィンドウを生成するウィンドウのハンドル
hWndOverlay	オーバーレイウィンドウのハンドルを取得するアドレス
x	オーバーレイウィンドウの位置(左)
y	オーバーレイウィンドウの位置(上)
nWidth	オーバーレイウィンドウの幅(最大 **720**)
nHeight	オーバーレイウィンドウの高さ(最大 **NTSC 480**、**PAL 576**)

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- オーバーレイウィンドウ生成時は、サイズが制限されますが、生成後は **DEC_Resize_Overlay_Window** 行により制限以上のサイズを指定できます。
- デコード中は生成できません。
- MVR-D2200V を使用している場合はオーバーレイウィンドウが作成できない場合、GDI で描画をおこなうためのウィンドが作成されます。

参照

DEC_Resize_Overlay_Window

DEC_Can_VideoPort_Overlay_Window

ビデオポートを利用したオーバーレイウィンドウを使用できるかどうか判断する。

書式

DEC_RETURN DEC_Can_VideoPort_Overlay_Window(UINT dec_id, HANDLE hMonitor, UINT nWidth, UINT nHeight);

引数

dec_id	デコーダの識別子
hMonitor	モニタハンドル
nWidth	オーバーレイウィンドウの幅
nHeight	オーバーレイウィンドウの高さ

戻り値

オーバーレイウィンドウが使用できる場合	**DEC_SUCCESS**
オーバーレイウィンドウが使用できない場合	**DEC_FAIL**

補足

- **hMonitor** には、マルチモニタでのプライマリまたはセカンダリのモニタハンドルを設定します。マルチモニタでない場合は、**hMonitor** に **NULL** を設定します。

DEC_Create_VideoPort_Overlay_Window

ビデオポートを利用したオーバーレイウィンドウを生成する。

書式

DEC_RETURN DEC_Create_Overlay_Window(UINT dec_id, HWND hWndParent, HWND* hWndOverlay, int x, int y, UINT nWidth, UINT nHeight);

引数

dec_id	デコーダの識別子
hWndParent	オーバーレイウィンドウを生成するウィンドウのハンドル
hWndOverlay	オーバーレイウィンドウのハンドルを取得するアドレス
x	オーバーレイウィンドウの位置（左）
y	オーバーレイウィンドウの位置（上）
nWidth	オーバーレイウィンドウの幅（最大 **720**）
nHeight	オーバーレイウィンドウの高さ（最大 **NTSC 480**、**PAL 576**）

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- オーバーレイウィンドウ生成時は、サイズが制限されますが、生成後は **DEC_Resize_Overlay_Window** 行により制限以上のサイズを指定できます。
- デコード中は生成できません。
- 生成した後は通常のオーバーレイウィンドウと同様に使用できます。
- MPL-D2000 でしか使用できません。

参照

DEC_Resize_Overlay_Window

DEC_Destroy_Overlay_Window

オーバーレイウィンドウを破棄します。

書式

DEC_RETURN DEC_Destroy_Overlay_Window(UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

●デコード中は破棄できません。

DEC_Move_Overlay_Window

オーバーレイウィンドウを移動します。

書式

DEC_RETURN DEC_Move_Overlay_Window(UINT dec_id, INT x, INT y);

引数

dec_id	デコーダの識別子
x	オーバーレイウィンドウの左上隅のスクリーン **x** 座標
y	オーバーレイウィンドウの左上隅のスクリーン **y** 座標

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Resize_Overlay_Window

オーバーレイウィンドウのサイズを変更します。

書式

DEC_RETURN DEC_Resize_Overlay_Window(UINT dec_id, UINT nWidth, UINT nHeight);

引数

dec_id	デコーダの識別子
nWidth	オーバーレイウィンドウの幅
nHeight	オーバーレイウィンドウの高さ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Show_Overlay_Window

オーバーレイウィンドウの表示状態を切り換えます。

書式

DEC_RETURN DEC_Show_Overlay_Window(UINT dec_id, BOOL fShow);

引数

dec_id デコーダの識別子
fShow オーバーレイウィンドウの表示／非表示

値	説明
TRUE	表示
FALSE	非表示

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

DEC_Get_Overlay_Window

オーバーレイウィンドウのハンドルを取得します。

書式

DEC_RETURN DEC_Get_Overlay_Window(UINT dec_id, HWND* pWnd);

引数

dec_id	デコーダの識別子
pWnd	オーバーレイウィンドウのハンドルを取得するアドレス

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Get_Overlay_Rect

オーバーレイウィンドウの表示領域を取得します。

書式

DEC_RETURN DEC_Get_Overlay_Rect(UINT dec_id, LPRECT pRect);

引数

dec_id	デコーダの識別子
pRect	オーバーレイウィンドウの表示領域を取得する **RECT** 構造体へのポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

● MVR-D2200V では使用できません。

DEC_Set_Overlay_Rect

オーバーレイウィンドウの表示領域を設定します。

書式

DEC_RETURN DEC_Set_Overlay_Rect(UINT dec_id, LPRECT pRect);

引数

dec_id	デコーダの識別子
pRect	オーバーレイウィンドウの表示領域を設定した **RECT** 構造体へのポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

● **pRect** で設定可能な範囲は次のとおりです。

ビデオフォーマット	水平位置	垂直位置
NTSC	**0 ～ 720**	**0 ～ 480**
PAL	**0 ～ 720**	**0 ～ 576**

● 再生時の画像サイズにより次の水平／垂直拡大率をかけて計算した値を設定します。

ビデオフォーマット	水平拡大率	垂直拡大率
NTSC SIF 以下	×**2**	×**2**
NTSC Half-D1 以下	×**2**	×**1**
NTSC 上記以外	×**1**	×**1**
PAL SIF 以下	×**2**	×**2**
PAL Half-D1 以下	×**2**	×**1**
PAL 上記以外	×**1**	×**1**

● デコード中は設定できません。

● MVR-D2200V では使用できません。

DEC_Start_Monitor

モニタを開始します。

書式

DEC_RETURN DEC_Start_Monitor(UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- **DEC_Start_Monitor** を実行すると、オーバーレイウィンドウは生成時のサイズに戻ります。
- デコード中は開始できません。

DEC_Stop_Monitor

モニタを停止します。

書式

DEC_RETURN DEC_Stop_Monitor(UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

●デコード中は停止できません。

DEC_Get_Monitor_Status

モニタの状態を取得します。

書式

DEC_RETURN DEC_Get_Monitor_Status(UINT dec_id, UINT* monitor);

引数

dec_id デコーダの識別子
monitor モニタの状態を取得するアドレス

値	説明
TRUE	モニタ中である
FALSE	モニタ中でない

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

DEC_Get_BSR_Parameter

メモリ転送用パラメータを取得します。

書式

DEC_RETURN DEC_Get_BSR_Parameter(UINT dec_id,
DEC_BSR_PARAMETER* pParam);

引数

dec_id	デコーダの識別子
pParam	メモリ転送用パラメータを取得する **DEC_BSR_PARAMETER** 構造体のポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Initialize_Ex

DEC_Set_BSR_Parameter

メモリ転送用パラメータを設定します。

書式

DEC_RETURN DEC_Set_BSR_Parameter(UINT dec_id,
DEC_BSR_PARAMETER* pParam);

引数

dec_id	デコーダの識別子
pParam	メモリ転送用パラメータを設定する **DEC_BSR_PARAMETER** 構造体のポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Initialize_Ex

DEC_Get_Overlay_Parameter

オーバーレイ表示パラメータを取得します。

書式

DEC_RETURN DEC_Get_Overlay_Parameter(UINT dec_id,
DEC_OVERLAY_PARAMETER* pParam);

引数

dec_id デコーダの識別子
pParam オーバーレイ表示パラメータを取得する **DEC_OVERLAY_PARAMETER** 構造体のポインタ

DEC_OVERLAY_PARAMETER 構造体の定義

オーバーレイ表示パラメータ情報が格納される。

```
typedef struct {
        UINT    nOverlayBrightness;
        int     nOverlayContrast;
        int     nOverlaySaturation;
} DEC_OVERLAY_PARAMETER;
```

nOverlayBrightness 明るさの値が格納される。

値	説明
DECMIN_VIDEO_OVERLAY_BRIGHTNESS	最小値
DECMAX_VIDEO_OVERLAY_BRIGHTNESS	最大値
DECDEF_VIDEO_OVERLAY_BRIGHTNESS	デフォルト値

nOverlayContrast コントラストの値が格納される。

値	説明
DECMIN_VIDEO_OVERLAY_CONTRAST	最小値
DECMAX_VIDEO_OVERLAY_CONTRAST	最大値
DECDEF_VIDEO_OVERLAY_CONTRAST	デフォルト値

nOverlaySaturation 色の濃さの値が格納される。

値	説明
DECMIN_VIDEO_OVERLAY_SATURATION	最小値
DECMAX_VIDEO_OVERLAY_SATURATION	最大値
DECDEF_VIDEO_OVERLAY_SATURATION	デフォルト値

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Set_Overlay_Parameter

オーバーレイ表示パラメータを設定します。

書式

DEC_RETURN DEC_Set_Overlay_Parameter(UINT dec_id,
DEC_OVERLAY_PARAMETER* pParam);

引数

dec_id	デコーダの識別子
pParam	オーバーレイ表示パラメータを設定する **DEC_OVERLAY_PARAMETER** 構造体のポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Get_Overlay_Parameter

DEC_Get_Video_Parameter

ビデオパラメータを取得します。

書式

DEC_RETURN DEC_Get_Video_Parameter(UINT dec_id,
DEC_VIDEO_PARAMETER* Video_Param);

引数

dec_id デコーダの識別子
Video_Param ビデオパラメータを取得する **DEC_VIDEO_PARAMETER** 構造体のポインタ

DEC_VIDEO_PARAMETER 構造体の定義

ビデオパラメータ情報が格納される。

```
typedef struct {
        int         TV_System;
} DEC_VIDEO_PARAMETER;
```

TV_System 放送規格の種類が格納される。

値	説明
DEC_VIDEO_TV_SYSTEM_NTSC	**NTSC**
DEC_VIDEO_TV_SYSTEM_PAL	**PAL**

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

DEC_Set_Video_Parameter

ビデオパラメータを設定します。

書式

DEC_RETURN DEC_Set_Video_Parameter(UINT dec_id,
DEC_VIDEO_PARAMETER* Video_Param);

引数

dec_id	デコーダの識別子
Video_Param	ビデオパラメータを設定する **DEC_VIDEO_PARAMETER** 構造体のポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- モニタ中に **DEC_VIDEO_PARAMETER** 構造体の **TV_System** の値を変更する場合は、**DEC_Stop_Monitor** によりモニタを停止させてから **DEC_Set_Video_Parameter** により値を変更します。モニタを再開する場合は、この後に **DEC_Start_Monitor** を行ないます。

参照

DEC_Get_Video_Parameter, DEC_Start_Monitor , DEC_Stop_Monitor

DEC_Get_Audio_Parameter

オーディオパラメータを取得します。

書式

DEC_RETURN DEC_Get_Audio_Parameter(UINT dec_id,
DEC_AUDIO_PARAMETER* Audio_Param);

引数

dec_id デコーダの識別子
Audio_Param オーディオパラメータを取得する **DEC_AUDIO_PARAMETER** 構造体のポインタ

DEC_AUDIO_PARAMETER 構造体の定義

オーディオパラメータ情報が格納される。

```
typedef struct {
        BOOL        fOutputMute;
        DWORD       dwLeftOutputAtt;
        DWORD       dwRightOutputAtt;
} DEC_AUDIO_PARAMETER;
```

fOutputMute ミュート状態が格納される。

値	説明
TRUE	ミュート **ON**
FALSE	ミュート **OFF**

dwLeftOutputAtt 左の音量が格納される。

値	説明
DECMIN_AUDIO_OUTPUT_ATT	最小値
DECMAX_AUDIO_OUTPUT_ATT	最大値

dwRightOutputAtt 右の音量が格納される。(値は **dwLeftOutputAtt** と同じ)

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

DEC_Set_Audio_Parameter

オーディオパラメータを設定します。

書式

DEC_RETURN DEC_Set_Audio_Parameter(UINT dec_id,
DEC_AUDIO_PARAMETER* Audio_Param);

引数

dec_id	デコーダの識別子
Audio_Param	オーディオパラメータを設定する **DEC_AUDIO_PARAMETER** 構造体のポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Get_Audio_Parameter

DEC_Get_Decode_Parameter

デコードパラメータを取得します。

書式

DEC_RETURN DEC_Get_Decode_Parameter(UINT dec_id,
DEC_DECODE_PARAMETER* Decode_Param);

引数

dec_id デコーダの識別子
Decode_Param デコードパラメータを取得する **DEC_DECODE_PARAMETER** 構造体のポインタ

DEC_DECODE_PARAMETER 構造体の定義

デコードパラメータ情報が格納される。

```
typedef struct {
        BOOL    fTimestampUsage;
        BOOL    fRepeatOnStop;
        BOOL    fFrameRepeat;
        BOOL    fPanScan;
        BOOL    fFrameClip;
} DEC_DECODE_PARAMETER;
```

fTimestampUsage **MPEG1** システムストリーム/**MPEG2** プログラムストリームの再生時、ビットストリーム内のタイムスタンプ(**PTS/DTS**)を用いるか否かの値が格納される。

値	説明
TRUE	タイムスタンプを用いる
FALSE	タイムスタンプを用いない

fRepeatOnStop デコード終了時、最終フレームを表示するか否かの値が格納される。

値	説明
TRUE	表示する
FALSE	表示しない

fFrameRepeat デコード終了時、最終フレームを表示する場合の表示方法が格納される。

値	説明
TRUE	フレーム表示
FALSE	フィールド表示

fPanScan アスペクト比 **16:9** の画像で、表示領域サイズの情報がある場合に表示領域の部分を拡大し、アスペクト比 **16:9** で表示するかの可否が格納される。

値	説明
TRUE	拡大表示する
FALSE	拡大表示しない

fFrameClip 再生時、表示フレームの端の部分を表示しないようにするかの可否が格納される。

値	説明
TRUE	表示フレームの端の部分を表示しない
FALSE	表示フレームの端の部分を表示する

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

補足

- **DEC_DECODE_PARAMETER** 構造体については、今後必要に応じてパラメータが拡張される場合があります。

DEC_Set_Decode_Parameter

デコードパラメータを設定します。

書式

DEC_RETURN DEC_Set_Decode_Parameter(UINT dec_id,
DEC_DECODE_PARAMETER* Decode_Param);

引数

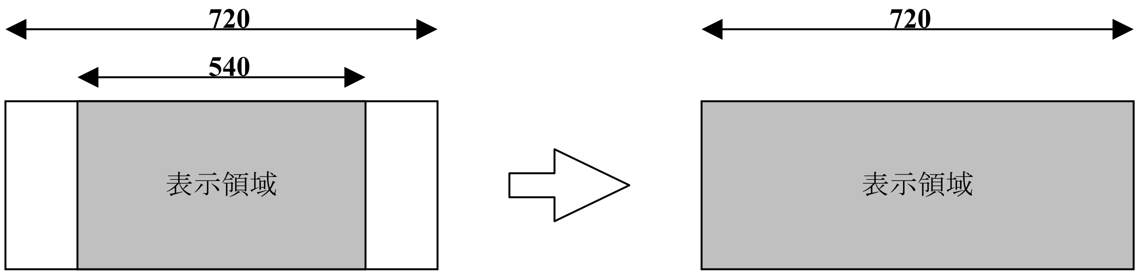
dec_id	デコーダの識別子
Decode_Param	デコードパラメータを設定する **DEC_DECODE_PARAMETER** 構造体のポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- **DEC_DECODE_PARAMETER** 構造体の **fPanScan** のに **TRUE** を設定すると、表示領域サイズの情報が画像データに含まれている場合、表示領域の部分を拡大し、アスペクト比 **16:9** で表示します。
  例えば、画像サイズ **720×480**、表示領域サイズ **540×480** の場合は下図のように表示されます。



● **DEC_DECODE_PARAMETER** 構造体の **fPanScan** のに **TRUE** を設定した場合、水平／垂直拡大率は次のようになります。

ビデオフォーマット	水平拡大率	垂直拡大率
NTSC　SIF 以下	×**8/3**	×**2**
NTSC　Half-D1 以下	×**8/3**	×**1**
NTSC　上記以外	×**4/3**	×**1**
PAL　SIF 以下	×**8/3**	×**2**
PAL　Half-D1 以下	×**8/3**	×**1**
PAL　上記以外	×**4/3**	×**1**

● **fPanScan** のに **FALSE** を設定した場合、または、表示領域サイズの情報が画像データに含まれていない場合、次の水平／垂直拡大率になります。

ビデオフォーマット	水平拡大率	垂直拡大率
NTSC　SIF 以下	×**2**	×**2**
NTSC　Half-D1 以下	×**2**	×**1**
NTSC　上記以外	×**1**	×**1**
PAL　SIF 以下	×**2**	×**2**
PAL　Half-D1 以下	×**2**	×**1**
PAL　上記以外	×**1**	×**1**

●**MPEG1** については、**fPanScan** は無視されます。

● **DEC_DECODE_PARAMETER**　構造体の次のメンバは MVR-D2200V では設定できません。
fTimestampUsage（常時　**TRUE**）、**fFrameRepeat(**常時　**TRUE)**、**fPanScan(**常時　**FALSE)**、**fFrameClip**（常時 **FALSE**）

参照

DEC_Get_Decode_Parameter

DEC_Play

デコードを開始します。

書式

DEC_RETURN DEC_Play(UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

●デコードは、**DEC_Seek** で指定した位置から開始します。

参照

DEC_Seek

DEC_Play_From

指定した位置からデコードを開始します。

書式

DEC_RETURN DEC_Play_From(UINT dec_id , DWORD dwPosition);

引数

dec_id	デコーダの識別子
dwPosition	再生開始位置（単位:ミリ秒）

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Seek

DEC_Pause

デコードを一時停止します。

書式

DEC_RETURN DEC_Pause (UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Resume

デコードを再開します。

書式

DEC_RETURN DEC_Resume (UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Stop

デコードを停止します。

書式

DEC_RETURN DEC_Stop (UINT dec_id);

引数

dec_id	デコーダの識別子

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Get_BitmapBits

静止画(BMP)をバッファで取得します。

書式

DEC_RETURN DEC_Get_BitmapBits(UINT dec_id, LPBYTE* lpByte, LPDWORD pdwLen, MVRSTILLCAPPARAM pParam);

引数

dec_id	デコーダの識別子
lpByte	データを格納するバッファへのポインタを格納するポインタ
pdwLen	取得した静止画のバイト数を格納するポインタ
pParam	静止画キャプチャパラメータを設定する **MVRSTILLCAPPARAM** 構造体

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- キャプチャ時に映像の DMA の転送を 1 度止めるので PC 上のオーバーレイの映像が一瞬停止します。(外部モニタへ出力されている映像は停止しません)
- 取得できる静止画のサイズはソースの画像サイズになります。
- 画像の縦のサイズが 480 以下の場合 **MVRSTILLCAPPARAM** 構造体の nFrameCap の値は何を指定しても MVRSTILL_FRAME になります。

DEC_Save_DIB

静止画(BMP)をファイルに保存します。

書式

DEC_RETURN DEC_Save_DIB(UINT dec_id, LPCTSTR lpszFileName, MVRSTILLCAPPARAM pParam);

引数

dec_id	デコーダの識別子
lpszFileName	保存するファイル名
pParam	静止画キャプチャパラメータを設定する **MVRSTILLCAPPARAM** 構造体

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Get_BitmapBits の補足を確認してください。

DEC_Save_JPEG

静止画(JPEG)をファイルに保存します。

書式

DEC_RETURN DEC_Save_JPEG(UINT dec_id, LPCTSTR lpszFileName, UINT nQuality, MVRSTILLCAPPARAM pParam);

引数

dec_id	デコーダの識別子
lpszFileName	保存するファイル名
nQuality	**JPEG** の圧縮率(**0**〜**9** の範囲で指定できます)を指定します。 **0 :** 約 **1/5** **1 :** 約 **1/7** **2 :** 約 **1/10** **3 :** 約 **1/12** **4 :** 約 **1/15** **5 :** 約 **1/17** **6 :** 約 **1/20** **7 :** 約 **1/25** **8 :** 約 **1/30** **9 :** 約 **1/40**
pParam	静止画キャプチャパラメータを設定する **MVRSTILLCAPPARAM** 構造体

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Get_BitmapBits の補足を確認してください

DEC_Get_Repeat

リピート状態を取得します。

書式

DEC_RETURN DEC_Get_Repeat(UINT dec_id, UINT* repeat);

引数

dec_id	デコーダの識別子
repeat	リピート状態を取得するアドレス

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Set_Repeat

DEC_Set_Repeat

リピート状態を設定します。

書式

DEC_RETURN DEC_Set_Repeat(UINT dec_id, UINT repeat);

引数

dec_id デコーダの識別子
repeat リピート状態を設定する

値	説明
TRUE	リピート **ON**
FALSE	リピート **OFF**

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

補足

- リピートの設定が有効になるのは、ファイルによるデコードの場合です。
- メモリ転送によるデコードの場合、リピートの設定を行なうことはできますが、実際にはリピートされません。

参照

DEC_Initialize_Ex, DEC_Get_Media

DEC_Get_Movie_File

デコードを行なうファイル名を取得します。

書式

DEC_RETURN DEC_Get_Movie_File(UINT dec_id, LPTSTR dec_file);

引数

dec_id	デコーダの識別子
dec_file	デコードを行なうファイル名を取得するアドレス

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Set_Movie_File

デコードを行なうファイル名を設定します。

書式

DEC_RETURN DEC_Set_Movie_File(UINT dec_id, LPCTSTR dec_file);

引数

dec_id	デコーダの識別子
dec_file	デコードを行なうファイル名を設定するアドレス

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- メモリ転送の場合は、設定できません。
- **DEC_Set_Movie_File** を実行後、デコード開始位置は先頭になります。

参照

DEC_Initialize_Ex

DEC_Get_Image_Size

デコードを行なうファイルの画像サイズを取得します。

書式

DEC_RETURN DEC_Get_Image_Size(UINT dec_id,
UINT* image_width, UINT* image_height);

引数

dec_id	デコーダの識別子
image_width	画像サイズ(幅)を取得するアドレス
image_height	画像サイズ(高さ)を取得するアドレス

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Get_Frame_Count

デコードを行なったフレーム数を取得します。

書式

DEC_RETURN DEC_Get_Frame_Count(UINT dec_id, UINT* frame_count);

引数

dec_id	デコーダの識別子
frame_count	フレーム数を取得するアドレス

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

●オーバーレイ表示中にオーバーレイウィンドウの位置・サイズを変更すると、フレーム数がカウントされなくなります。

DEC_Get_Time

デコードを行なった時間を取得します。

書式

DEC_RETURN DEC_Get_Time(UINT dec_id, double* time);

引数

dec_id	デコーダの識別子
time	時間を取得するアドレス(単位:秒)

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- メモリ転送によるデコードの時は、オーバーレイ表示中にオーバーレイウィンドウの位置・サイズを変更すると、時間が更新されなくなります。

DEC_Get_Type

デコードの種類を取得します。

書式

DEC_RETURN DEC_Get_Type(UINT dec_id, DEC_TYPE* type);

引数

dec_id	デコーダの識別子
type	デコードの種類を取得する **DEC_TYPE** のポインタ

値	説明
DEC_TYPE_UNKNOWN	不明
DEC_TYPE_MPEG1_AUDIO	**MPEG1** オーディオ
DEC_TYPE_MPEG1_VIDEO	**MPEG1** ビデオ
DEC_TYPE_MPEG1_SYSTEM_STREAM	**MPEG1** システムストリーム
DEC_TYPE_MPEG2_AUDIO	**MPEG2** オーディオ
DEC_TYPE_MPEG2_VIDEO	**MPEG2** ビデオ
DEC_TYPE_MPEG2_PROGRAM_STREAM	**MPEG2** プログラムストリーム

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

DEC_Get_File_Type

ファイルの種類を取得します。

書式

DEC_RETURN DEC_Get_File_Type(UINT dec_id, LPCTSTR dec_file, DEC_TYPE* type);

引数

dec_id	デコーダの識別子
dec_file	ファイル名のアドレス
type	デコードの種類を取得する **DEC_TYPE** のポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

参照

DEC_Get_Type

DEC_Get_Playback_Time

再生時間を取得します。

書式

DEC_RETURN DEC_Get_Playback_Time(UINT dec_id, LPDWORD lpdwTime);

引数

dec_id	デコーダの識別子
lpdwTime	再生時間を取得するためのバッファへのポインタ(単位:ミリ秒)

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- **DEC_Set_Movie_File** で指定したファイルについての再生時間を取得します。
- **DEC_Seek** により再生位置の指定が可能なファイルついて再生時間を取得することができます。

参照

DEC_Set_Movie_File, DEC_Seek

DEC_Seek

再生を開始する位置を指定します。

書式

DEC_RETURN DEC_Seek(UINT dec_id, DWORD dwPosition);

引数

dec_id	デコーダの識別子
dwPosition	再生開始位置（単位:ミリ秒）

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

● 再生位置の指定に関しては、次の制限事項があります。

① プログラム・ストリーム／ビデオ・エレメンタリ・ストリームについては、ビデオ・エレメンタリ・ストリーム内の GOP 内のタイム・コードを利用する。よって、次に当てはまるファイルに関しては再生位置を指定できない。

- GOP がない
- タイム・コードが設定されていない
- 再生時間が 24 時間より大きい（タイム・コードは 24 時間までしか格納できないため）

② GOP の時間間隔（通常 0.4～1.0 秒）より小さい時間間隔の位置への移動はできない。

③ Video CD ファイル（RIFF CDXA 形式の.dat ファイル）については、再生位置を指定できない。

● 再生位置に関しては、指定された位置を超えない範囲で、最も近い位置が再生開始位置になります。

① プログラム・ストリーム／ビデオ・エレメンタリ・ストリーム・ファイルの場合は、MPEG ビデオ GOP の最初のピクチャ表示時刻が、指定された時刻を超えない範囲で、最も近いものをスキャンし、その位置を返す。

② オーディオ・エレメンタリ・ストリーム・ファイルの場合は、指定された時刻を超えない範囲で、最も近いオーディオ・アクセス・ユニット（フレーム）の先頭位置を返す。

DEC_Get_Sync_Stc_Value

HD814210 の **SYNC STC** レジスタの値を取得します。

書式

DEC_RETURN DEC_Get_Sync_Stc_Value(UINT dec_id, LPDWORD pdwValue);

引数

dec_id	デコーダの識別子
pdwValue	**HD814210** の **SYNC STC** レジスタの値を取得する変数へのポインタ

戻り値

成功	**DEC_SUCCESS**
失敗	**DEC_FAIL**

補足

- デコーダ・コア・レジスタからの読み込み時の（デコーダ・コア・アクセス・レジスタを用いた）ハンドリングは **MvrAvc.dll** 内部で実行されるので、呼出側は、他のレジスタと同様に、単にデコーダ・コア・レジスタを指定するだけで良い。
- この **API** は通信サンプルプログラムでのみ使用しています。
- MVR-D2200V では使用できません。

DEC_Get_IRE_Setting

NTSC セットアップレベル（黒レベル）の設定を取得します。

書式

DEC_RETURN DEC_Get_IRE_Setting(UINT dec_id, LPINT lpnIreSetting);

引数

dec_id デコーダの識別子
lpnIreSetting セットアップレベルの情報を格納するための **INT** 変数へのポインタ

値	説明
MVR_IRE_LEVEL_0_NTSC	**0 IRE**
MVR_IRE_LEVEL_75_NTSC	**7.5 IRE**

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

DEC_Set_IRE_Setting

NTSC セットアップレベル（黒レベル）を設定します。

書式

DEC_RETURN DEC_Set_IRE_Setting(UINT dec_id, INT nIreSetting);

引数

dec_id デコーダの識別子
nIreSetting 設定するセットアップレベル

値	説明
MVR_IRE_LEVEL_0_NTSC	**0 IRE**
MVR_IRE_LEVEL_75_NTSC	**7.5 IRE**

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

補足

● 日本では 0 IRE が標準です。

DEC_Get_Overlay_Parameter_Ex

オーバーレイの設定を取得します。

書式

DEC_RETURN DEC_Get_Overlay_Parameter_Ex(UINT dec_id,
DECPARAM_OVERLAY fProp, LPDWORD lpdwValue);

引数

dec_id デコーダの識別子
fProp 情報を取得する対象を指定します。以下の値のいずれかを指定します。

値	意味
DECPARAM_OVERLAY_BRIGHTNESS	明るさの値を取得します。
DECPARAM_OVERLAY_HUE	色合いの値を取得します。
DECPARAM_OVERLAY_CONTRAST	コントラストの値を取得します。
DECPARAM_OVERLAY_SATURATION	色の濃さの値を取得します。
DECPARAM_OVERLAY_FLIP	フリッピングの設定状態を取得します。
DECPARAM_OVERLAY_BOB	**BOB** の設定状態を取得します。

lpdwValue 取得した値を格納するための **DWORD** 変数へのポインタ

戻り値

成功 **DEC_SUCCESS**
失敗 **DEC_FAIL**

補足

- MVR-D2200V 以外では使用できません。

参照

DEC_Set_Overlay_Parameter_Ex、**DEC_Get_Overlay_Parameter_Range_Ex**

DEC_Set_Overlay_Parameter_Ex

オーバーレイの設定を行ないます。

書式

DEC_Set_Overlay_Parameter_Ex(UINT dec_id, DECPARAM_OVERLAY fProp, DWORD dwValue);

引数

dec_id デコーダの識別子

fProp 情報を設定する対象を指定します。以下の値のいずれかを指定します。

値	意味
DECPARAM_OVERLAY_BRIGHTNESS	明るさの値を取得します。
DECPARAM_OVERLAY_HUE	色合いの値を取得します。
DECPARAM_OVERLAY_CONTRAST	コントラストの値を取得します。
DECPARAM_OVERLAY_SATURATION	色の濃さの値を取得します。
DECPARAM_OVERLAY_FLIP	フリッピングの設定状態を取得します。
DECPARAM_OVERLAY_BOB	**BOB** の設定状態を取得します。

dwValue 設定する値

戻り値

成功 **DEC_SUCCESS**

失敗 **DEC_FAIL**

補足

- MVR-D2200V 以外では使用できません。
- 設定できる範囲等の情報は **DEC_Get_Overlay_Parameter_Range_Ex** で取得します。

参照

DEC_Get_Overlay_Parameter_Ex、**DEC_Get_Overlay_Parameter_Range_Ex**

DEC_Get_Overlay_Parameter_Range_Ex

オーバーレイ表示パラメータの範囲を取得します。

書式

DEC_RETURN DEC_Get_Overlay_Parameter_Range_Ex(UINT dec_id,
DECPARAM_OVERLAY fProp, LPDECPARAM_OVERLAY_RANGE lpstRange);

引数

enc_id エンコーダの識別子
fProp 情報を取得する対象を指定します。以下の値のいずれかを指定します。

値	意味
DECPARAM_OVERLAY_BRIGHTNESS	明るさの値を取得します。
DECPARAM_OVERLAY_HUE	色合いの値を取得します。
DECPARAM_OVERLAY_CONTRAST	コントラストの値を取得します。
DECPARAM_OVERLAY_SATURATION	色の濃さの値を取得します。
DECPARAM_OVERLAY_FLIP	フリッピングの設定状態を取得します。
DECPARAM_OVERLAY_BOB	**BOB** の設定状態を取得します。

lpstRange 情報を取得する **DECPARAM_OVERLAY_RANGE** 構造体へのポインタ

DECPARAM_OVERLAY_RANGE 構造体の定義

オーバーレイ表示パラメータの範囲の情報が格納される。

```
typedef struct {
    BOOL     fEnable;
    DWORD    dwMin;
    DWORD    dwMax;
    DWORD    dwDef;
} DECPARAM_OVERLAY_RANGE, *LPDECPARAM_OVERLAY_RANGE;
```

fEnable 使用可能の情報が格納される。
dwMin 範囲の最小の値が格納される。
dwMax 範囲の最大の値が格納される。
dwDef デフォルトの値が格納される。

戻り値

成功 **ENC_SUCCESS**
失敗 **ENC_FAIL**

補足

● MVR-D2200V 以外では使用できません。

参照

DEC_Get_IRE_Setting、DEC_Set_IRE_Setting

DEC_Get_Last_Error

最新のエラー情報を取得します。

書式

ULONG DEC_Get_Last_Error(UINT dec_id);

引数

dec_id デコーダの識別子

戻り値

ULONG エラーコード（**APPENDIX** 1. エラーコード表参照）

補足

- **DEC_Get_Last_Error** を実行すると、エラー情報はクリアされます。エラーが発生していない場合や新たにエラーが発生するまでは、**MVR_ERROR_SUCCESS** が返されます。

APPENDIX

1. エラーコード一覧

Mvrapi.dll 使用時のエラーコードは次の通りです。

MVR_ERROR_SUCCESS

正常に終了。

MVR_ERROR_PENDING

指定された操作はまだ完了していない。

MVR_ERROR_SYSTEM_ERROR

システムに関するエラーが発生した。

MVR_ERROR_NOT_PENDING

指定された操作は完了している。

MVR_ERROR_INVALID_ASYNCOP

イベントハンドルが無効。

MVR_ERROR_UNSUPPORTED

サポートされていないファンクションを実行した。

MVR_ERROR_INVALID_TYPE

セッションの種類が無効。

MVR_ERROR_INVALID_OPTION

セッションのオプションが無効。

MVR_ERROR_INVALID_ADDRESS

アドレスが無効。

MVR_ERROR_LIMIT_EXCEEDED

制限値を越えている。

MVR_ERROR_INVALID_HANDLE

ハンドルが無効。

MVR_ERROR_INVALID_STREAM

ストリームが無効。

MVR_ERROR_INVALID_BUFFER

バッファが無効。

MVR_ERROR_INVALID_MESSAGE

メッセージが無効。

MVR_ERROR_CANCELED

指定された要求は取り消された。

MVR_ERROR_BUSY

指定されたイベントハンドルは現在使用されている。

MVR_ERROR_BUFFER_ERROR

バッファの初期化に失敗した。

MVR_ERROR_MESSAGE_ERROR

メッセージに関するエラーが発生した。

MVR_ERROR_BAD_PARAM

無効なパラメータを指定した。

MVR_ERROR_HARDWARE_ERROR

ハードウェアに関するエラーが発生した。

MVR_ERROR_ALLOC_FAILURE

リソースの割り当てに失敗した。

MVR_ERROR_INVALID_EVENT

イベントが無効。

MVR_ERROR_INVALID_STATE

ステータスが無効。

MVR_ERROR_CURRENT_STATE

ストリームが既に指定された状態にある。

MVR_ERROR_INVALID_FRAME_TYPE

フレームの種類が無効。

MVR_ERROR_INSUFFICIENT_BUFFERS

バッファサイズが不十分。

MVR_ERROR_TIMEOUT

タイムアウトが発生した。

MVR_ERROR_INVALID_CODEC_STATE

コーデックの状態が無効。

MVR_ERROR_INVALID_FORMAT

ストリームの形式が無効。

MVR_ERROR_INVALID_OPERATION

指定された操作は無効。

MVR_ERROR_OVERFLOW

オーバーフローが発生した。

MVR_ERROR_UNDERFLOW

アンダーフローが発生した。

MVR_ERROR_SESSION_CANCELED

セッションが取り消された。

MVR_ERROR_UCODE_NOT_FOUND

マイクロコードが見つからない。

MVR_ERROR_AUDIO_UCODE_NOT_FOUND

オーディオのマイクロコードが見つからない。

MVR_ERROR_PARSE_ERROR

ビットストリーム形式が無効。

MVR_ERROR_END_OF_STREAM

シーク操作中にビットストリームの最後に達した。

MVR_ERROR_DISK_FULL

ディスクがいっぱいになった。

MVR_ERROR_STREAMS_CLOSED

ストリームは破棄されている。

MVR_ERROR_INTERNAL_ERROR

内部エラーが発生した。

MVR_ERROR_INVALID_SHORT_BUFFER

バッファは完全に満たされていない。

MVR_ERROR_NO_VIDEO_SOURCE

ビデオ信号が入力されていない。

MVR_ERROR_DEVICE_NOT_FOUND

デバイスが見つからない。

MVR_ERROR_EVENT_DATA_OVERRUN

MPEGビデオ・データまたはオーディオ**PCM**データの、デバイスからの収集(吸い上げ)が間に合わない。

（原因）

1. **PC**の処理性能が低い。

2. 録画中に、他のアプリケーションを起動/動作させたため、データの吸い上げが一時的に間に合わなくなった。

3. 正しいビデオ信号が入力されていない。

ドライバ内部のマルチプレクサ・プログラムは、データ吸い上げのため一定個数のバッファを割り当て、そのバッファを**MPEG**ビデオ・データまたはオーディオ**PCM**データの吸い上げに使い回します。入力されているビデオ信号の周期が乱れたり、フィールドが欠落したりすると、それに応じて、**MPEG**ビデオ・エンコーダから出力されるデータの転送周期が不正確となり、ビデオ・データに対してマルチプレクサが割り当てているバッファが足らなくなったり、あるいは、逆に、オーディオ・データにバッファを割り当てられなくなったりします。
具体的には、下記の様な状況で発生します。
(a) 劣化したビデオ・テープを入力ソースとして使用すると、ビデオ信号が正しい周期(**NTSC**ならば、**29.97Hz**)で入力されていない場合があります。
(b) ビデオ・テープに複数の映像ソースが重ね書きされていると、映像ソースの変わり目で、不正なビデオ信号が発生します。
(c) 入力映像が途中で途切れると、不正なビデオ信号が発生する場合があります。

MVR_ERROR_EVENT_DATA_UNDERRUN

アンダーランが発生した。

MVR_ERROR_EVENT_DEVICE_ERROR

デバイスエラーが発生した。

MVR_ERROR_EVENT_LOSS_OF_VIDEO

データの引き抜きは間に合っているが、引き抜かれたデータが不正、またはデータの一部が欠落した。

MVR_ERROR_EVENT_EVENT_LOSS

イベントエラーが発生した。

MVR_ERROR_EVENT_UNRECOVERABLE_ERROR

回復不可能なエラーが発生した。
システムを一度終了してから、再起動させてください。

MVR_ERROR_EVENT_WARNING

警告エラーが発生した。

MVR_ERROR_EVENT_VIDEO_ENCODER_VBV_UNDERFLOW

ビット・レート・コントロールに失敗した。
ビットレートをあげるか、ビデオ入力フォーマットを SIF または Half-D1 に設定してください。

MVR_ERROR_EVENT_VIDEO_ENCODER_OVERRUN_STATE

データの引き抜きが間に合わなかった等の原因で、ビデオ入力の数フレームがエンコードできなかった。
原因となるのは、

(**a**) 録画中に他のアプリを起動した。あるいは、他のアプリで重い処理を実行した。

（**b**） 他の**I/O(**特に、**PCI**バスを使用する**I/O)**に時間がかかっている。

（**c**） ドライバ関連のモジュールがスワップ・アウトされている。

（**d**） マシンの性能が低い。

カノープス株式会社
本社／〒651-2241　神戸市西区室谷 1-2-2（神戸ハイテクパーク内）
●　MVR Series Development Kit の
技術的なお問い合わせは<カノープスシステム開発サポート>へ
インターネット　E-mail　：mvrsdk@canopus.co.jp
FAX　：078-992-4203
※FAX でお問い合わせの際は、FAX 番号をよくお確かめください。

☆最新の情報をホームページでご覧になれます。

http://www.canopus.co.jp/